Solaris ISP Server 2.0 Administration Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
U.S.A.

Part No: 805-7695–10
February 1999

Adobe PostScript

Please
Recycle

Contents

Figures

Preface

Solaris ISP Server™ (formerly known as Solaris for ISPs) is a suite of software that provides the best possible platform and subscriber services on the Solaris operating system for Internet Service Providers. The Solaris ISP Server 2.0 Administration Guide provides information about Solaris ISP Server™ platform and service software. It provides an overview of the software and examines the use of the directory services, the schema extension, and integrating new applications into the product.

Who Should Read This Book

The audience for this book includes system administrators: anyone who maintains a multi-user computer system running Solaris ISP Server.

How This Book Is Organized

This book contains the following chapters:

Chapter 1 discusses the features of Solaris ISP Server platform and service software.

Chapter 2 discusses generic Internet security risks, the structure of Solaris ISP Server in relation to them, and measures you can take to safeguard your network against those risks.

Chapter 3 describes the required directory structure and the access control rules.

Chapter 4 supplies step-by-step instructions for creating required directory entries for domains and NAS objects, both at the command line and by using the Deja tool.

Chapter 5 describes the procedures for creating directory entries for the end-user, covering command line, Deja, and bulk loading of entries.

Chapter 6 provides reference material on the object classes and attributes added to the Sun™ Directory Services schema. It also discusses steps for maintaining the schema, for restoring the schema, and avoiding schema dependence in your code.

Chapter 7 examines the requirements for integrating existing service applications with Sun™ Internet Administrator™.

Appendix A describes the steps for configuring the DNS, creating required UNIX accounts, and making needed directory entries for a virtually-hosted domain in Solaris ISP Server.

Related Documentation

Documentation related to Solaris ISP Server includes printed manuals, AnswerBooks, PostScript, PDF, and man pages. Each is listed below.

Manuals

The documents listed here are provided in printed version. These manuals are also available as AnswerBooks™ and as on line help files on the CD.

- *Solaris ISP Server 2.0 Administration Guide*
- *Solaris ISP Server 2.0 Installation Guide*
- *Solaris ISP Server 2.0 Reference Guide*

AnswerBook

These documents are provided as AnswerBook™ on the CD.

- *Solaris ISP Server 2.0 Administration Guide*
- *Solaris ISP Server 2.0 Installation Guide*
- *Solaris ISP Server 2.0 Reference Guide*
- *Sun Directory Services 3.1 Administration Guide*
- *Sun Directory Services 3.1 User's Guide*
- *SunScreen SKIP User's Guide, Release 1.1*

PostScript and PDF

These documents are provided as PostScript™ and PDF on the CD:

- *Solaris ISP Server 2.0 Administration Guide*
- *Solaris ISP Server 2.0 Installation Guide*
- *Solaris ISP Server 2.0 Reference Guide*

Man Pages

The locations of Solaris ISP Server platform and service software man pages are listed below:

- The Sun™ Internet Administrator™ man pages are installed to two locations. Those pertaining to commands on the Sun Internet Administrator host machine are installed at `/opt/SUNWixamc/man`. Those pertaining to commands on the service host machine are installed at `/opt/SUNWisp/man`.
- The host configuration and LDAP access API man pages are located in `/opt/SUNWisp/man`.
- The Sun™ WebServer™ man pages are located in `/usr/share/man`.
- The Sun™ Internet FTP Server™ man pages are located in `/opt/SUNWixfta/1.1/man`.
- The Sun™ Internet News Server™ man pages are located in `/opt/SUNWsns/man`.
- The man pages for the Network Cache Accelerator are located in `/usr/share/man`
- The Sun™ Directory Services man pages are located in `/opt/SUNWconn/man`.
- The FLEXLm man pages are located in `/opt/SUNWste/license_tools/man`.
- The SunScreen™ SKIP man pages are located in `/opt/SUNWicg/man`.
- The JDK™ man pages are located in `/usr/share/man`.

Online Help

The majority of application-specific information is provided in the form of online help for easy access while working with the software. Each software component has its own help set, accessible from the graphical user interface.

The `README` *File*

The Solaris ISP Server `README.html` file is a short file in the installation media that contains late breaking news, bugs, release information, and pointers to software readme files and documents for installing. It is located in the root directory of the installation media.

Other Related Documents

You may also want to consult the following books related to the subject matter discussed in this book.

- *Solaris Advanced Installation Guide*
- *Automating Solaris Installations (A Custom JumpStart Guide)*, by Paul Anthony Kasper and Alan L. McClellan, SunSoft Press, 1995.
- *Computer Security Policies and SunScreen ™ Firewalls*, by Kathryn M. Walker and Linda Croswhite Cavanaugh, Prentice Hall, 1998.

Definitions of Typefaces

The following table describes the typographic changes used in this book.

Typeface or Symbol	**Meaning**	**Example**
`AaBbCc123`	The names of commands, files, and directories	Edit your `.login` file. Use `ls -a` to list all files.
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*
AaBbCc123	Book Titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.

CHAPTER **1**

Introducing Solaris ISP Server

Welcome to Solaris ISP Server™ 2.0 software. This software suite builds upon the open architecture and scalability of the Solaris operating system to provide the optimum operating environment for Internet service providers and their customers.

This chapter introduces the software in Solaris ISP Server. It contains the following sections:

- "Solaris ISP Server Overview" on page 2 describes a typical Solaris ISP Server installation and its operation.
- "Solaris ISP Server Platform Software" on page 5 describes the platform software that comes with the product.
- "ISP Services" on page 17 describes the ISP services that are a part of Solaris ISP Server.

What is Solaris ISP Server?

Solaris ISP Server has two categories of software. The platform software extends the Solaris operating system. It adds functionality that enables ISP services and supports the ISP in deploying those services, but it is not directly accessed by ISP subscribers. The ISP services provide subscriber functionality such as Internet news, FTP, and World Wide Web access while taking advantage of the extended Solaris environment.

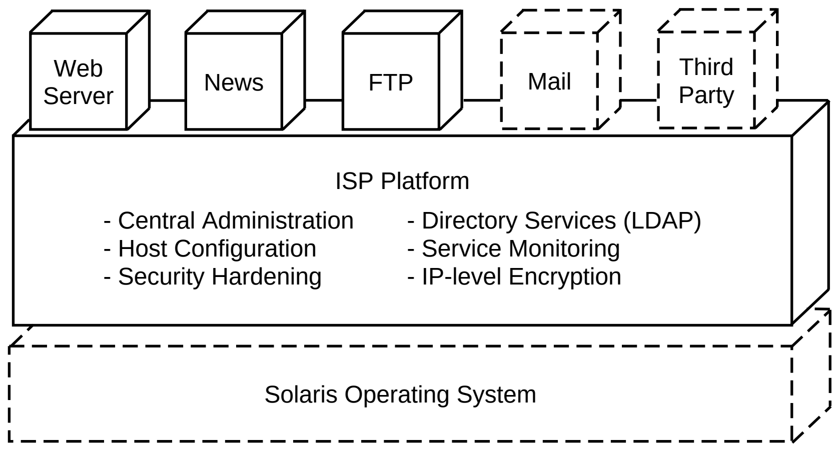


Figure 1–1 Solaris ISP Server Builds on the Solaris Operating System

Solaris ISP Server Overview

Solaris ISP Server is designed to run on a network; although you can install the entire product on a single machine, the result would not be functional in a working ISP environment. For best performance and security results, we recommend segregation of services by host machine, in a configuration appropriate for your planned use. The product presumes the existence of standard primary and secondary DNS (domain name service) servers, but requires no special configuration of them.

The directory services are integral to Solaris ISP Server. The product presumes a single installation of Sun™ Directory Services managing all subscriber, administrator, and service configuration data for the system. Sun Internet Administrator and some of the services do not work in the absence of a directory. When set up to interoperate with Sun Internet Mail Server, certain special configuration steps are required to keep the two directories (Solaris ISP Server's directory server and that contained within the mail product) in synchronization. The sections that follow describe these networks in a general manner and describe the flow of basic operations.

A Basic Solaris ISP Server Installation

In the simple example that follows, some services share host computers. This is not a requirement; segregate services by host machine if that is the best configuration for your system.

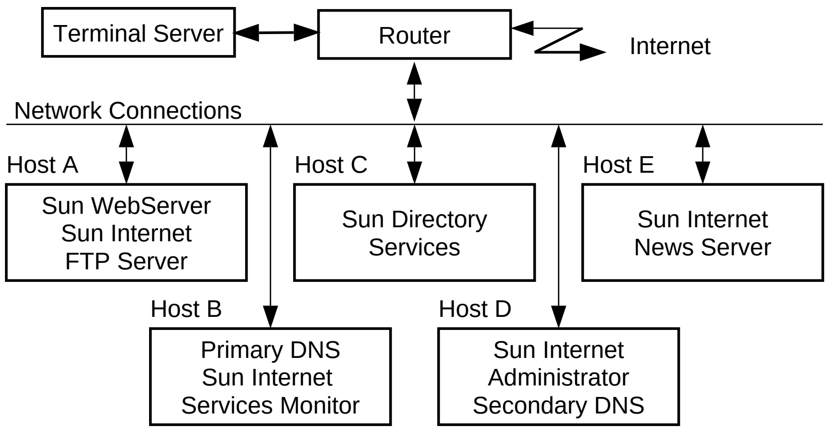


Figure 1–2 Basic Solaris ISP Server Network

Host C can also be the RADIUS server, if desired. RADIUS is a part of Sun Directory Services and is configured for authentication against the directory.

A Solaris ISP Server Installation with Sun Internet Mail Server

When Solaris ISP Server is set up to interoperate with Sun Internet Mail Server, the mail server must be installed on a different computer from Sun Directory Services.

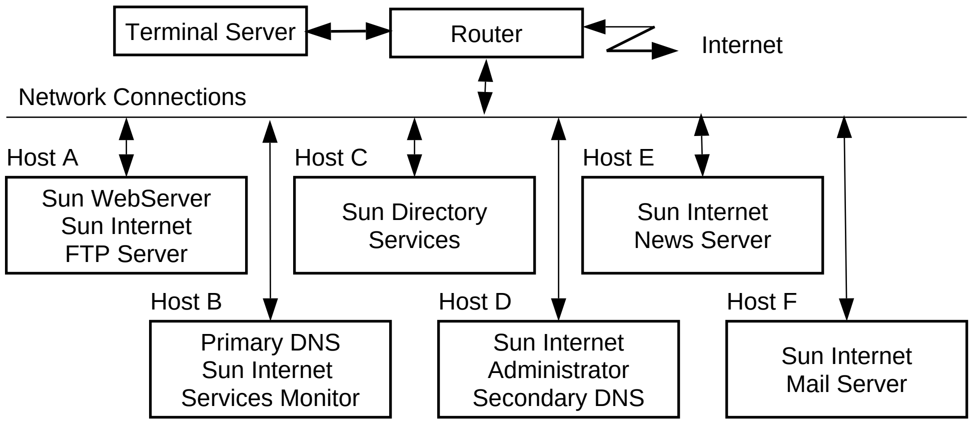


Figure 1–3 Solaris ISP Server Network with Sun Internet Mail Server

In this setup, the directory in the mail server is configured as a slave to the main Sun Directory Services. All user information management is performed on the main directory on Host C. User information is replicated across to Host F.

Solaris ISP Server Operation

Solaris ISP Server user information is stored in the directory, and user authentication is performed using LDAP. For example, if the RADIUS server packaged with Sun Directory Services is used, the server binds to the directory and searches the user entries for the user name and `authsuffixname` (a directory attribute defining the user's domain) provided by the user. Once the user's directory entry is found, the server compares the password provided with the one in the directory, validating the user. Once authenticated, the user communicates with the desired service using the appropriate protocol.

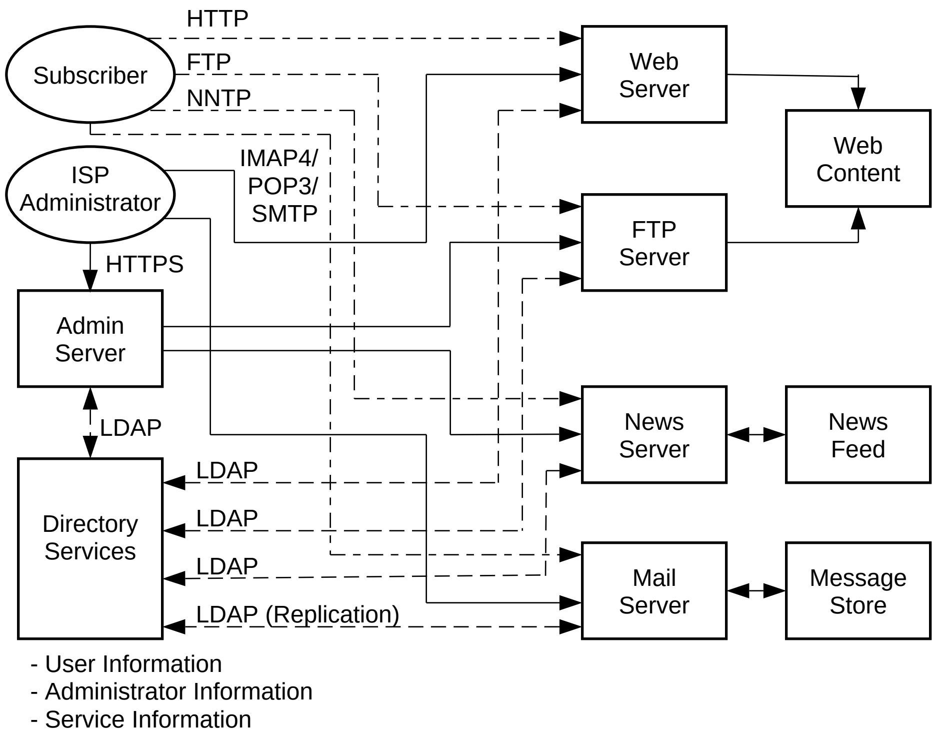


Figure 1–4 Communications and Protocols in Solaris ISP Server

The ISP administrator typically accesses the administration server for Sun Internet Administrator. Once Sun Internet Administrator authenticates the administrator against the directory, it passes the login information to the services (single sign-on) as the administrator accesses them. Three-tier services invisibly authenticate the user against the directory again, blocking any attempt by an intruder pretending to be a Sun Internet Administrator. Three-tier services are described in "Three-Tier Service Architecture" on page 6.

Solaris ISP Server Platform Software

Solaris ISP Server includes the following enhancements to the Solaris operating system:

- Sun™ Internet Administrator™
- Host configuration software
- Security Hardening
- Sun™ Internet Services Monitor™
- Sun™ Directory Services
- FlexLM License Server
- SunScreen™ SKIP
- HotJava™ browser
- Java™ Development Kit

Sun Internet Administrator Overview

Sun Internet Administrator provides secure central management for distributed ISP services. It gives ISPs the following benefits:

- *Single sign-on for administrators.* ISP administrators log in to Sun Internet Administrator once to access all functions for which they have authorization. Applications in the three-tier configuration and managed from Sun Internet Administrator receive login information from it; the user is not subsequently challenged. Three-tier services are described in "Three-Tier Service Architecture" on page 6.
- *Administrator access control.* Access is controlled per ISP service. An administrator allowed to manage FTP servers on the network may or may not also have access to news servers. Console administrators (those who can manage Sun Internet Administrator processes) have access to all managed services.
- *Secure communications between administrators' client machines and service hosts.* Web server access control lists (ACLs) protect Sun Internet Administrator from access by unauthorized users. SSL can be used on the HTTP connection. Also, the optional SunScreen SKIP software can be installed and configured on all connections to the Sun Internet Administrator, and from it to the service host machines, encrypting those communications.
- *Logging of administrator actions for traceability.* Each administrator action, from initial login attempt through logout, is logged via the `syslog` utility. This provides both troubleshooting and accountability information.

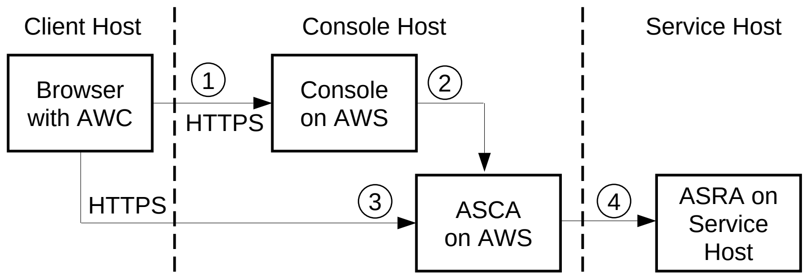
- *Remote management of ISP services.* Services provided with Solaris ISP Server can all be managed from Sun Internet Administrator, regardless of their location on the network. Additionally, Sun™ Internet FTP Server™, Sun™ Internet News Server™, and Sun™ Directory Services are three-tier components and receive all the security benefits built into Sun Internet Administrator. See "Three-Tier Service Architecture" on page 6 and "Two-Tier Service Architecture" on page 8 for more information on service interaction with Sun Internet Administrator.
- *Extensibility for existing services.* ISPs can integrate their own applications with Sun Internet Administrator and manage them in the same way as services provided with Solaris ISP Server. See Chapter 7 for instructions on integrating applications with Sun Internet Administrator.

Sun Internet Administrator supports services in two architectures: three-tier and two-tier. Only the three-tier architecture receives all of the listed security benefits. Four types of application interfaces are supported:

- *Three-tier, browser-based* applications receive all security benefits offered by Sun Internet Administrator.
- *Two-tier, browser-based* applications cannot make use of the single sign-on feature, but are manageable through the Sun Internet Administrator. If they use Sun WebServer to support the administration application, they can configure it to provide administrator authentication and access control, but not single sign-on. The two-tier architecture is included to support existing applications without requiring additional programming. See Chapter 7 for details on this configuration.
- *X-based* applications receive all the benefits of a three-tier application.
- *Command-line* functions (scripts, programs, or in combination) receive all the benefits of three-tier applications. Any number of them can be registered for a given service and managed by Sun Internet Administrator, which constructs a web interface to the command-line programs.

Three-Tier Service Architecture

The recommended three-tier browser-based application architecture receives all Sun Internet Administrator security benefits.



Legend

AWC: Administration Web Client. That part of the Sun Internet Administrator GUI that is downloaded to the administrator's browser.

AWS: Administration Web Server. The web server that serves the Sun Internet Administrator GUI and any ASCAs present for managed services.

ASCA: Administration Server Client Agent. The portion of a managed service's user interface coresident with the AWS.

ASRA: Administration Server Remote Agent. The portion of a managed service's user interface coresident with the service.

Figure 1–5 Three-Tier ISP Service Architecture

As shown in Figure 1–5, an administrator uses the following steps to access a service's administration functions:

1. From a browser, the administrator accesses either `http://`*<hostname>*`:50080/ispmc` or `https://`*<hostname>*`:50087/ispmc` (the location of the main Sun Internet Administrator GUI page).

   The AWC is downloaded to the client browser, and the administrator chooses a service to manage.

2. Sun Internet Administrator prompts the administrator for user name and password. The administrator need not use a UNIX account for access to the user interface; a directory services repository (Sun Directory Services) manages administrator information for Sun Internet Administrator. This connection should be secured by using secure HTTP.

   The selected service resolves to a URL, designating the services's ASCA. The server agent GUI is downloaded to the administrator's browser in response. At this step, control passes to the service's administration program.

3. Subsequent access is directly between the client browser and the application's server agent on the AWS.

   The AWS authenticates the administrator against the directory services, and logs each administrator request via `syslog`. If the administrator has appropriate access, requests are passed to the ASCA. If not, access is denied and a log entry is made.

4. The ASCA communicates with the ASRA via a protocol independent of Sun Internet Administrator (chosen by the developer of the service). Appropriate IP-level security measures should be taken to protect this connection and its traffic.

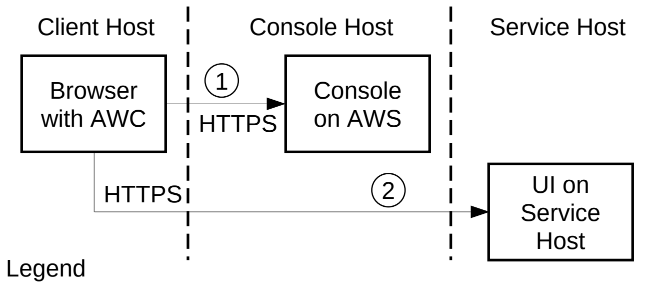
The ASRA again authenticates and logs each administrator action.

To secure the communications for three-tier applications, we recommend using SSL or SunScreen™ SKIP on the client browser connection and SunScreen SKIP on all other intercomputer connections.

ASCA and ASRA modules for command-line and X-based programs are provided in Solaris ISP Server. Sun Internet Administrator uses them automatically when you register these applications.

Two-Tier Service Architecture

For some applications, especially existing services, a two-tier architecture for access via Sun Internet Administrator is more practical. These services can be managed from Sun Internet Administrator, but do not receive the security benefits of single sign-on and central logging (though they can do their own logging in `syslog`).



Legend

AWC: Administration Web Client. That part of the Sun Internet Administrator GUI that is downloaded to the administrator's browser.

AWS: Administration Web Server. The web server that serves the Sun Internet Administrator GUI.

Figure 1–6 Two-Tier ISP Service Architecture

As shown in Figure 1–6, an administrator uses the following steps to access a service's administration functions:

1. From a browser, the administrator accesses either `http://`*<hostname>*`:50080/ispmc` or `https://`*<hostname>*`:50087/ispmc` (the location of the main Sun Internet Administrator GUI page).

   This step is the same as for the three-tier architecture. The AWC is downloaded to the client browser, where the administrator can choose a service to manage.

   The selected service resolves to a URL, designating the component's user interface.

2. Subsequent access is directly between the client browser and the service's remote agent. Appropriate IP-level security measures should be taken to protect this connection and its traffic.

In a two-tier architecture, services are not able to take advantage of the single sign-on feature. If a two-tier web-based application uses Sun WebServer to support its user interface, it can configure the web server to provide the same service-level access protection as a three-tier application enjoys. See Chapter 7 for information on this configuration.

To secure the communications for a two-tier application, we recommend using SSL or SunScreen SKIP.

Restoring the Administration Web Server Configuration

Sun Internet Administrator uses an instance of Sun WebServer to support its web-based user interface. This web server is referred to as the administration web server (AWS). You can, reconfigure the AWS to suit your requirements, for example to use SSL for security reasons.

Refer to the Sun WebServer online help to reconfigure the AWS. In particular, see `httpd.conf(4)` and the Sun WebServer on line help for configuring SSL. The web server instance that is the AWS is called “aws” in the Sun WebServer user interface.

To ensure that you do not lose the default configuration, this section discusses the location of the default AWS configuration files and the method to restore the default settings.

Backups of the AWS default configuration files are located in `/etc/opt/SUNWixamc/awsconf/default/*`. The files in use are at `/var/opt/SUNWixamc/awsconf`. To restore the default settings:

```
cp /etc/opt/SUNWixamc/awsconf/default/*.*
/var/opt/SUNWixamc/awsconf/.
```

Ensure that `adm` has read and write access to all files.

Note - For the effective functioning of Sun Internet Administrator, *do not* change the default settings in `aws.conf`, `site.conf`, `map.conf`, `realms.conf`, and `access.acl`.

Host Configuration Software Overview

The Solaris ISP Server host configuration software provides the following functionality:

- *Software installation.* Administrators install, uninstall, and upgrade all Solaris ISP Server software using the host configuration software. Administrators can save installation scenarios for use in a JumpStart™ finish script to repeat installations automatically.
- *Solaris foundation security hardening.* To improve security and conserve resources, unneeded Solaris services are disabled. Security-related components of Solaris are configured appropriately for an ISP environment.

- *Intrusion detection.* Periodically, the intrusion detector checks its log file, determining whether any failed log-on attempts have occurred since the last check. If an intrusion attempt has occurred, the detector collects the logged data and passes it to the user-specified notification mechanism (such as electronic mail). The period for the intrusion check is configurable.
- *Log file management.* Audit and `syslog` logs are cycled daily. The log file management daemon archives logs weekly and deletes any archive older than one month. See the `hclfmd(1m)` man page for details.
- *Server process management.* This `cron` job ensures that server processes (such as news servers) are indeed running. If any server has stopped abnormally, the server process manager starts that server.

How Solaris ISP Server Installs

Because the typical UNIX server must run a variety of applications, the default Solaris installation assumes that most UNIX services are needed. ISPs focus more narrowly on providing specific services in a public environment. They have heavy performance and security requirements.

To configure Solaris to their needs, ISP administrators typically perform elaborate hardening tasks. They disable unneeded Solaris services and change file permissions to close security vulnerabilities. This process can take hours.

The host configuration software in Solaris ISP Server automates this hardening process for the administrator. In addition to copying the necessary software packages to their proper locations, it hardens the underlying Solaris foundation, changing file owners and modes where appropriate as well as configuring Solaris security and logging mechanisms. The final step in this process is selectively disabling standard Solaris services (such as `finger` or `rlogin`) when they do not support the purpose of a given host machine. The administrator controls which services are disabled.

Solaris ISP Server host configuration can be performed interactively by using its graphical user interface, or repeatably and non-interactively using JumpStart.

Host Configuration Model

The configuration process works by building a scenario of the current state of the system, what software components are available to be installed, and what the user has selected for install or uninstall.

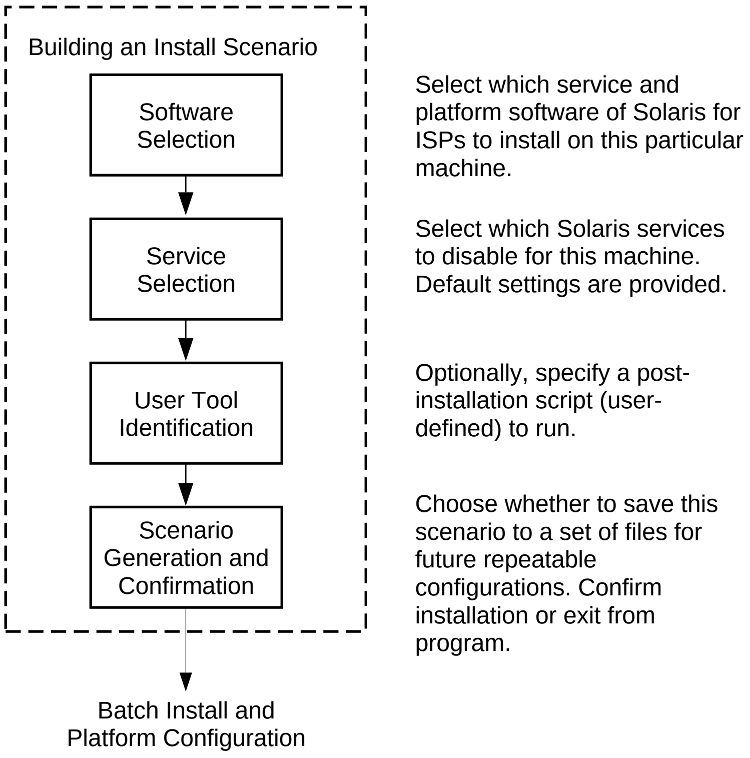

Building an Install Scenario
Software Selection
Service Selection
User Tool Identification
Scenario Generation and Confirmation
Batch Install and Platform Configuration
Select which service and platform software of Solaris for ISPs to install on this particular machine.
Select which Solaris services to disable for this machine. Default settings are provided.
Optionally, specify a post-installation script (user-defined) to run.
Choose whether to save this scenario to a set of files for future repeatable configurations. Confirm installation or exit from program.

Figure 1–7 Solaris ISP Server Host Configuration Process

The host configuration software can also be used to reconfigure a host after installation, adding and removing services as needed.

Repeatable Configuration

Interactive host configuration (using the graphical user interface) provides the option to save a configuration scenario (in the form of a binary and some associated files). By creating and saving a scenario, the ISP administrator can use it in a JumpStart finish script, forming a non-interactive, one-step installation. Such JumpStart installations are repeatable and can be used to configure multiple machines identically.

JumpStart is a part of the Solaris operating system that can perform customized, repeatable installations of Solaris both locally and remotely. See the *Solaris Advanced Installation Guide* for details on how to create a custom JumpStart installation. See the *Solaris ISP Server Installation Guide* and the `hcjump(1M)` man page for information on how to use a scenario file in a finish script for a custom JumpStart installation.

Log File Management and Intrusion Detection

The host configuration software includes a resident daemon, `hclfmd`, that performs log file management. This daemon runs as root. It starts at boot time and performs the following functions:

- It parses the list of log files in `/etc/syslog.conf` for file paths that do not start with `/dev` (files associated with system devices) and performs a cleanup, journal, and cycle pass every day.

  For every log file written by `syslogd`, it performs the following functions:

  - It renames the existing log file and creates a new daily log.
  - It sends the restart signal `(-HUP)` to the syslog daemon to create a new daily log.
  - It generates a weekly archive by compressing daily log files every week and stores it as *name.YYYYMMDD-YYYYMMDD*`.tar.Z`.
  - It discards weekly archives that are more than a month old.

- It obtains the location of audit logs from `/etc/security/audit_control` and performs a cleanup, journal, and cycle pass every day.

  It performs the following functions for every locally mounted audit directory:

  - It executes `audit -n` to create a new daily log. This signals the audit directory to close the current audit file and open a new audit file in the current audit directory.
  - It generates a weekly archive by compressing daily log files every week and stores it as `audit.`*YYYYMMDD-YYYYMMDD*`.tar.Z`.
  - It discards weekly archives that are more than a month old.

- It performs an intrusion detection check on the interval set by the user. For details, see `hclfmd(4)`.

  - It detects and reports every failed authorization entry in `syslog` files.
  - By default, `/etc/opt/SUNWisp/hc/hclfmd.conf` is configured to send mail to root for every failed authorization attempt entered in `syslog`.

Note - You can reconfigure this file. By default, it is configured as follows:
`/var/log/badauth:/usr/bin/mailx -s ''`*%f*`'' root <` *%c* where:

- `/var/log/badauth` is the file where the entries are made.
- `/usr/bin/mailx -s` is the command to send mail to root.
- "*%f*" is the subject-line of the mail, containing the name of the file where the entries were detected.
- "*%c*" is the new content of the `syslog` file.

Security Hardening

This Solaris ISP Server component can be installed to ensure security for passwords and to safeguard file permissions to the file owner. The functionality of this unit is similar to the functionality of the script in `ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz`.

- This component, when installed, runs a script that make modes of files installed as part of Solaris packages more secure. These changes are as follows:
  - Removes group and world read permissions for `setuid` and `setgid`.
  - Removes group and world write permissions on all non-`setuid` files that meet any of the following criteria:
    - The file has group and world readable permission, but no world writable permission.
    - The file has world executable permission.
    - The file has identical owner, group, and world permissions.
    - It is a bin-owned directory or nonvolatile file and has identical group and world read and executable permissions.
  - Removes write permissions for owners on executables not owned by root.
- It adds `umask 077` to `/.cshrc`, `/.profile`, and `/.zshenv`. This makes the default file permission for files created under an interactive root shell readable and writable only by root. If you do not want this umask, add a umask of your choice to these files prior to installing Solaris ISP Server. The configuration script will respect your settings.
- It adds root to `/etc/ftpusers` to disable root's ability to connect to the host using FTP.
- It sets noshell as the default shell for `sys, uucp, nuucp,` and `listen` accounts to log unauthorized logging attempts. This makes it easier to detect intrusion on the system.
- It sets `MAXWEEKS=12` in `/etc/default/passwd`. If local files are used for password management, this forces all passwords to change periodically.
- It creates `S35umask` to make default file permission for files created by system daemons writable only by the file owner.
- It disables a denial of service attack by adding the lines
  - `ndd -set /dev/ip ip_respond_to_echo_broadcast 0`
  - `ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0`

in the file `/etc/rc2.d/S69inet`.

- It replaces `/etc/syslog.conf` with a new version for ensuring more granular logging and for detecting intrusion. This new version isolates messages by both facility and logging level and sends the high-level messages to a central logging server.

- It executes `bsmconv` and configures `/etc/security` to log administrative actions, and logins and logouts. This enables C2 auditing, which may catch events missed by `syslog`.
- All changes made by this unit are logged to `/var/sadm/install/contents`. This enables patch installation in the future.

Sun Internet Services Monitor

The performance monitoring software allows an ISP to set up client machines that emulate a subscriber's experience with the ISP services. The performance monitoring applet can be set to connect to any combination of web, mail, news, and directory services and collect information on their performance from a subscriber's perspective. This data is collected on the monitoring server machine and viewable with a web browser.

Sun Internet Services Monitor is a two-tier application. It is manageable through Sun Internet Administrator, but does not receive the benefits of single sign-on or administrator authentication. See "Two-Tier Service Architecture" on page 8 for more information on the two-tier architecture.

Sun Directory Services

This Lightweight Directory Access Protocol (LDAP) implementation provides a shared repository for user, administrator, and service configuration information. Features in this release of Sun Directory Services include:

- Conformance to LDAP v3 Internet standards
- A Remote Access Dialup User Service (RADIUS) server that provides authentication for remote users connecting to the network through a Network Access Server (NAS).
- A Network Information System (NIS) server that integrates into an existing NIS environment to provide an integrated naming service
- A complete suite of administration tools, including the Deja directory editor, a Java-implemented administration console for management of the directory, and a web gateway for access from any Java-enabled browser

Sun Directory Services is manageable from Sun Internet Administrator as an X-based application (three-tier).

Sun Directory Services allows one thousand entries in the directory before requiring a license. A license certificate for five thousand entries ships with Solaris ISP Server and must be redeemed and registered with the FlexLM license server before it takes effect. See the instructions in the *Solaris ISP Server Installation Guide* for details of redeeming and installing the license certificate. Additional licenses are available from your sales representative.

See Chapter 3 and Chapter 6 of this book for information about the role of Sun Directory Services in Solaris ISP Server.

FlexLM License Server

The FlexLM license server is used by Sun Directory Services to manage licenses of various sizes. If you already have a license server in your network (version 4.1), you can use it to serve Sun Directory Services licenses. Otherwise, install the software provided.

Sun Directory Services allows one thousand entries before requiring a license. This is sufficient to install and initialize the directory. A five thousand entry license is supplied with Solaris ISP Server. Follow the directions in the *Solaris ISP Server Installation Guide* for acquiring a license key and configuring the server.

ISP Directory Information API

This API provides C and Java programming language access to the directory services. The functions return information specific to the directory information tree (DIT) used by Solaris ISP Server.

LDAP Client Library

The LDAP client library is an implementation of the LDAP v3 standard. It provides support for client applications communicating with an LDAP server such as Sun Directory Services.

Sun Internet Mail Server Interoperability Support

This set of configuration files and scripts installs with the Sun Directory Services component. Once the directory and Sun Internet Mail Server are installed on their respective machines, a series of configuration steps aligns the two directory stores and sets up the Sun Internet Mail Server directory as a replicant (slave) of the main directory services. User information is shared across the two directories, but managed centrally from the master directory.

For complete information on configuring SIMS and the directory for interoperation, see the *Solaris ISP Server 2.0 Installation Guide*.

SunScreen SKIP

SunScreen™ SKIP is based on the Simple Key Management for Internet Protocols (SKIP) standard of key management for IP encryption. Characteristics of SunScreen SKIP include:

- Automatic certificate exchanges
- Sessionless protocols
- Multicast and unicast packet protocols for IPv4 and IPv6
- Certificate Discovery Protocol (CDP)

SunScreen SKIP provides 40-bit or 128-bit encryption, depending upon your local and United States export restrictions.

HotJava Browser

The HotJava browser is provided with Solaris ISP Server to support Sun Internet Administrator and other administration user interfaces in the product. It supports the following Internet standards and protocols:

- Java Development Kit
- HTTP/1.1
- HTML 3.2
- Tables and Frames
- Persistent Cookies
- GIF and JPEG Media Formats
- AU Audio Format
- FTP and Gopher File Transfer Protocols
- SMTP and MIME E-mail Protocols
- SOCKS Protocol
- Secure Sockets Layer (SSL) 3.0
- Java Archive (JAR) Format

Java Development Kit

The Java Development Kit (JDK) is provided with Solaris ISP Server to support the use of Java in the product. The Solaris JDK includes the following capabilities:

- Internationalization
- Signed applets

- JAR file format
- AWT (window toolkit) enhancements
- JavaBeans™ component model
- Networking enhancements
- Math package for large numbers
- Remote Method Invocation (RMI)
- Reflection
- Database connectivity (JDBC)
- Serialization PROTOCOL_VERSION_2 (read only)

ISP Services

Services in this version of Solaris ISP Server include:

- Sun™ WebServer™, with Network Cache Accelerator
- Sun™ Internet News Server™
- Sun™ Internet FTP Server™

Sun WebServer

Sun WebServer is a highly reliable, secure, standards-based web server for accessing, managing, and distributing information over the Internet, extranets, or intranets. Features in this release of Sun WebServer include:

- *Support for Java Servlet Development Kit 2.0.* Using servlets rather than CGI scripts provides the cross-platform advantages of Java on the server side while improving processing speed. JSDK 2.0 adds session support to its previous feature set.
- *Enhanced dynamic content performance.* This enhancement to the CGI interfaces allows content developers to use the Solaris doors interface to speed up interprocess communications.
- *Support for HTTP/1.1.* Sun WebServer supports the latest version of the hypertext transfer protocol, including named virtual hosts and content negotiation.
- *Enhanced scalability.* Multiple instances of the server process can run on a single machine and can be managed through the same administration graphical user interface. The number of available virtual hosts is thus raised geometrically.
- *Secure HTTP communications.* Sun WebServer includes secure-socket layer (SSL) functionality and support for Verisign certificates for safe, encrypted communications.

- *Microsoft FrontPage support.* Sun WebServer supports popular FrontPage extensions in the areas of authoring, administration, and dynamic content.

Sun WebServer is a two-tier application. It is manageable through the Sun Internet Administrator and is configured to share administrator data with it. See Chapter 7 for details on this configuration and “Two-Tier Service Architecture” on page 8 for more information on the two-tier architecture.

Sun WebServer can also be administered via its command-line functions.

Network Cache Accelerator

NCA is a kernel-loadable module that enhances HTTP performance for static content. It is supported on Solaris 7 and must be configured with a web server such as Sun WebServer 2.1.

The Network Cache and Accelerator, NCA, increases web server performance by maintaining an in-kernel cache of web pages accessed during HTTP requests. NCA provides full HTTP 1.1 support in the kernel by either handling the request or passing it to the web server for processing.

Sun Internet News Server

Sun Internet News Server is a high-performance, highly-scalable news server. Features include:

- *High performance and client-connection scalability.* A multithreaded, multiprocess daemon handles client connections, taking advantage of multiprocessor Solaris servers to provide scalability in handling large numbers of simultaneous news reader connections.
- *Full-featured news feed handling.* Based on INN from the Internet Software Consortium (INN release 1.5 sec. 2), Sun Internet News Server maintains all the usability improvements of that implementation in the area of news feed handling.
- *Separate feed handling and news reader service functionality.* The service for news readers is separable from the feed handling functions to enable horizontal scalability in handling news reader connections.
- *Centralized, browser-based management.* Sun Internet News Server is a three-tier service that integrates with Sun Internet Administrator for centralized administration and full security benefits. See “Three-Tier Service Architecture” on page 6 for more information on the three-tier architecture.

Sun Internet News Server can also be administered via its command-line functions.

Sun Internet FTP Server

This scalable, high-performance FTP server offers the following:

- *Multiple domains on a single host.* Sun Internet FTP Server supports IP-based virtual hosting. Each virtual host has its own configuration files that tune the server for that particular domain.
- *Configurable user authentication.* Sun Internet FTP Server can use the entries in the Sun Directory Services directory to authenticate users.
- *Centralized, browser-based management.* Sun Internet FTP Server is a three-tier service that integrates with Sun Internet Administrator for centralized administration and full security benefits. See "Three-Tier Service Architecture" on page 6 for more information on the three-tier architecture.

Sun Internet FTP Server can also be administered via its command-line functions.

CHAPTER 2

Ensuring Security

It is a truism of the Internet that the network is insecure. The very nature of being connected means that you do not completely control the data you process. Because requests can come from outside your network, you need to know who is making the request. Because your data can be routed through many outside networks, you need to protect your network's integrity and privacy. This chapter discusses the security concerns in a Solaris ISP Server™ installation, and how to maintain that security in a network connected to the Internet.

Specific information in this chapter includes:

- "Security Concepts" on page 21
- "Solaris ISP Server Structure and Security" on page 24
- "How To Tighten Security" on page 25

Security Concepts

This section provides an introduction to security concepts and terminology. It is not intended as a complete tutorial on the subject. Many good books have been written on the subject. Please see this book's for suggested reading on security.

Security Goals

The goals of a good security policy are simple to describe, if not to attain. They are to ensure that only authorized people gain access to the system, that communications on the network are kept private from outsiders, and that data being communicated is kept safe. The terms defined in this section are those commonly used to describe these security goals.

Authentication

Authentication is verifying that a person requesting access is who he or she claims to be. Knowing who is making a request for service, or is sending a piece of information, is crucial to the security of the ISP's system. Technology that provides authentication (such as what Sun™ Internet Administrator™ provides for administrators), assures the system that a message is coming from a trusted source, and not from an intruder pretending to be a legitimate user.

Any technology that requires a login before granting access is providing some user authentication. Signed certificates, which can be used with the SunScreen™ SKIP product, offer a strong form of authentication.

Authorization (Access Control)

Authorization ensures that a given (authenticated) person is allowed access to the requested system resources or data. Even legitimate users are not all allowed root access, for example.

Sun Internet Administrator provides access control, limiting the access of ISP administrators to only those services that they need to manage while doing their jobs.

Accountability

Accountability is the concept of knowing who performed what actions. This is not so much a security guarantee as a way of identifying the source of trouble and removing that user from the system. In Solaris ISP Server, accountability is achieved through a thorough logging of administrator actions. Be sure to examine (audit) the logs regularly to catch any questionable activity.

Privacy

Given the public nature of the Internet, the privacy of data on the wire is of prime importance. You need to know that an intruder cannot listen to communications and pick up user passwords to be used later to gain access to your systems. Data passed between computers must be protected from being read by unauthorized individuals.

Cryptography can ensure the privacy of network communications. The SSL technology in Sun WebServer provides encryption of information accessed over the Web. The SunScreen SKIP software provides encryption of all IP traffic between interfaces on your network.

Data Integrity

Integrity is knowing that the message sent by a legitimate user is identical to that received by the system, and that its content has not been altered along the way. Clearly this is a significant security requirement, given the potentially disastrous effects of an intruder adding dangerous code to the end of an innocuous message. These "Trojan horse" attacks can frequently be devastating.

Encryption of the data, particularly strong encryption coupled with signed certificate authentication, provides data integrity by making it practically impossible for an intruder to modify a legitimate user's message. SunScreen™ SKIP provides this technology in Solaris ISP Server.

Types of Attacks on Network Security

This section describes some of the most common types of attacks on a network connected to the Internet.

Denial of Service Attacks

These attacks make the system unavailable for customers, often by overloading the service with spurious requests. These attacks flood the network with useless traffic resulting in an inability to serve customers. Such attacks can crash the system in extreme cases, or just make the system extremely slow and thus unusable by customers.

Buffer Overrun Exploits

If a program does not check buffer limits when reading and receiving data, this opening can be exploited to add arbitrary data into a program or file system. When run, this data can give the intruder root access to your system.

Snooping and Replay Attacks

Snooping attacks involve an intruder listening to traffic between two machines on your network. If traffic includes passing unencrypted passwords, an unauthorized individual can potentially access your network and read confidential data.

IP Spoofing

In a spoofing attack, the intruder listening to your network traffic finds the IP address of a trusted host and sends messages indicating that the message is coming from that trusted host.

Internal Exposure

Many network break-ins are the result of a malicious or disgruntled present or former employee misusing access to information or breaking into your network.

Solaris ISP Server Structure and Security

This section discusses some aspects of Solaris ISP Server that introduce the potential for some security risks.

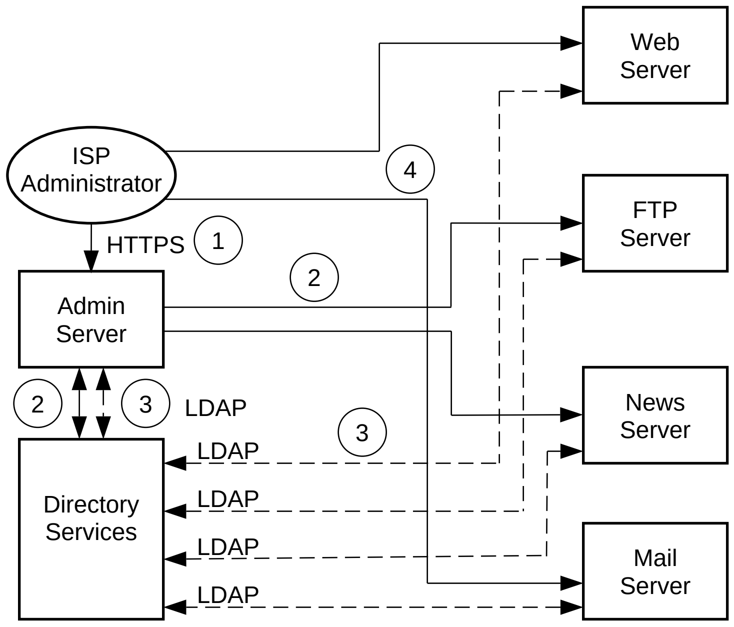


Figure 2–1 Areas to Protect in the Solaris ISP Server Network

The connections labeled with circled numbers in Figure 2–1 indicate network connections that should be protected for a secure Solaris ISP Server system.

1. The connection between an ISP administrator's browser and Sun Internet Administrator is the first region of vulnerability. The administrator's machine could be anywhere on the Internet, and thus its communications must be protected. The administrator's password, used when logging into Sun Internet Administrator, would allow an intruder snooping on the line to access the same services and cause major damage. To protect this connection, configure the administration web server (AWS) for Sun Internet Administrator to use SSL or SKIP. Instructions for configuring SSL are in the Sun™ WebServer™ online help.

Note - The web server instance that is the AWS is called "aws" in the Sun WebServer user interface.

---

2. The connections between the AWS and the applications it manages are also vulnerable to various snoop and spoof attacks. Sun Internet Administrator passes administrator login information to each application's administration interface. Should that information be picked up by an intruder, that intruder could log in as the administrator and cause damage.

   In addition, Sun Internet Administrator invokes command-line functions via its own remote execution mechanism. Should this mechanism be broken into, the intruder would have access to powerful command-line functions, some of which run as root. To protect against both of these risks, you should configure SunScreen SKIP (or its equivalent) on these communication paths. SKIP authenticates incoming traffic and encrypts data so that it cannot be viewed or modified by others while in transit.
3. Connections to the directory services are particularly important, as user and service information is passed along constantly, and all such information is stored in the directory. Sun™ Directory Services encrypts password values, but data is readily available to a snooper on the wire. To protect this information, you should configure SunScreen SKIP on these communication paths.
4. Connections between the administrator's machine and two-tier applications also require protection. You can configure an IP-level security mechanism, such as SunScreen SKIP between the two machines. In the case of Sun WebServer, you can configure the administrative user interface to use SSL. Refer to the Sun WebServer online help for information on configuring SSL. Because SSL uses a different port, remember to reregister the service with Sun Internet Administrator. Instructions for this are in the Sun Internet Administrator online help.

How To Tighten Security

This section discusses standard security precautions you can take to protect your network, the security mechanisms in Solaris ISP Server, and how to use these mechanisms in a Solaris ISP Server installation.

Standard Security Precautions

The following standard security precautions will help safeguard your network.

- Design your network for as few software components per machine as is compatible with the machine's purpose. Any software product has the potential to introduce security holes, whether through known vulnerabilities or through bugs. The fewer processes running and the fewer protocols supported, the more secure the computer will be.

  In addition, if an intruder were to compromise one computer, only a portion of the resources and services in your system would be involved.

- Disable Solaris services that are not needed on the particular computer. Solaris ISP Server host configuration software offers recommendations based on your choice of application software, for example, disabling some 'r' commands (such as `rlogin`) to ensure protection for passwords and to restrict access to hosts for unauthorized individuals. Unless you have a specific reason for enabling a service, accept these recommendations.
- Change passwords regularly and encourage using difficult-to-guess passwords. The directory services do not enforce periodic changing of passwords; you must have your users change them at appropriate intervals.
- Use public-key cryptography to encrypt all traffic between trusted hosts at the IP level. SunScreen SKIP, bundled with Solaris ISP Server, authenticates incoming IP traffic and ensures that outgoing data is not altered or viewed by others while in transit.
- Use routers that can identify trusted hosts and block spoofed IP addresses.
- Fix vulnerabilities and bullet proof your code. Ensure that all applications check buffer limits and prevent overruns.
- Grant access only to the portions of the system that employees need to do their jobs. Limit administrator rights to only those services they actually manage. Sun Internet Administrator supports this effort by offering a centralized way to manage administrator access. Administrators do not even need UNIX accounts to do their work.
- Implement security mechanisms such as network monitoring and firewalls.

Security Mechanisms in Solaris ISP Server

Security mechanisms in Solaris ISP Server include a hardening step performed during installation, access control lists (ACLs), encryption software (SunScreen SKIP), and intrusion detection based on examination of log files.

Authentication

Authentication in Solaris ISP Server is performed using a private Pluggable Authentication Module (PAM). User login information is validated against the directory using LDAP. This form of authentication allows for more users on a single system, and provides a better security model than standard UNIX authentication. It allows central authentication for users of all systems on the network and removes the need to create a UNIX account for every user.

Access Control

Sun Directory Services has ACLs that are set during installation. They are defined so that only appropriate entities can add, change, or delete entries. Specific information on the ACLs configured by Solaris ISP Server is in Chapter 3 of this book.

Sun WebServer has ACLs that protect web site content based on user-definable realms. All Solaris ISP Server web-based interfaces are protected by these ACLs, to prevent unauthorized access.

Hardening of the Solaris Operating System

The Solaris ISP Server host configuration software disables unnecessary Solaris services and changes the modes and owners of certain system files, reducing the potential security gaps in the system. This process is configurable; you control which services are disabled and which other hardening steps are performed.

Intrusion Detection

Using `syslog`, Solaris ISP Server applications log administrator's actions. These logs are checked at a configurable interval for login attempts. A user-specified alert is triggered if failed attempts are noted.

Secure Sockets Layer (SSL)

SSL provides authentication and authorization (access control) by the use of signed (or self-signed) certificates. Privacy of communications and data integrity are provided by cryptography. SSL encryption uses a 40-bit algorithm in the global version of the product and a 128-bit algorithm in the US/Canada version.

To configure SSL on Sun WebServer, refer to the Sun WebServer online help.

SunScreen™ SKIP

SunScreen SKIP provides cryptographic technology configurable at the IP level. It performs host-level authentication and access control by use of certificates and ACLs. It provides privacy by way of encryption (40-bit global, and 128-bit US/Canada), and data integrity by means of encryption.

To configure SunScreen SKIP between two machines, first use the Solaris ISP Server host configuration software to install it on both systems. Then, see the *SunScreen*™

SKIP User's Guide for instructions on creating certificates and further configuration steps.

CHAPTER 3

Using Directory Services

Solaris ISP Server™ uses Sun™ Directory Services to store subscriber data, store component software information, and login information for Sun™ Internet Administrator™ and for some services. This information includes standard object classes and newly-defined objects that support Solaris ISP Server functionality.

This chapter describes the Solaris ISP Server information tree structure, explains how to support multiple domains in the directory, and describes the directory access controls. For information on the schema extension itself, see Chapter 6.

Specific topics in this chapter include:

- "Sun Directory Services Administration Tools" on page 29
- "Solaris ISP Server Directory Structure" on page 30
- "Supporting Multiple Independent Domains" on page 33
- "Directory Services Access Control" on page 34

Sun Directory Services Administration Tools

Sun Directory Services provides the following tools for using and administering the directory:

- The Deja tool, a Java-based directory editor, provides add, modify, and delete capability.
- The Sun Directory Services administration console offers local and remote administration of the server.
- The Web gateway allows browse access from any browser.

- A complete set of command-line programs.

The Sun Directory Services books, *Sun Directory Services 3.1 Administration Guide* and *Sun Directory Services 3.1 User's Guide*, include complete information on starting and using these tools. They are provided in AnswerBook2 format on the product CD-ROM. The command-line programs are documented in man pages in section 1 (`/opt/SUNWconn/man`).

Solaris ISP Server Directory Structure

Solaris ISP Server requires a specific structure in the directory information tree (DIT), which is created during installation and configuration. Solaris ISP Server host configuration creates two naming contexts, referred to as the Open Systems Interconnection (OSI) tree and the domain component (DC) tree.

Each naming context is defined according to the data you provide during installation. Initially, two naming contexts are created. The OSI naming context is created directly from the distinguished name you provide (for example, o=*sun*,c=*us*). If you entered `sun.com` as your domain, the DC naming context is defined as dc=*com*, and an entry is made beneath that for dc=*sun*. The host configuration process also creates the requisite entries.

Portions of the two trees are parallel. This parallel structure facilitates mapping of domain names from a DNS request through the DC tree to the actual content entries in the OSI tree.

OSI Tree Structure

The OSI tree contains the actual entries for Solaris ISP Server, its component services, administrators of those services, and subscribers to the services. The required structure is shown in Figure 3–1.

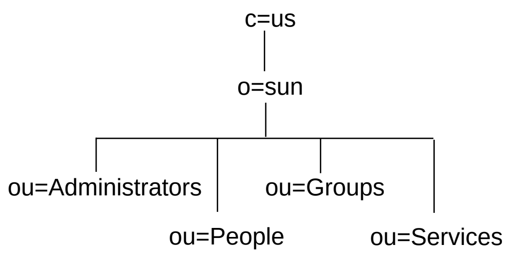


Figure 3–1 Solaris ISP Server OSI Tree

In the OSI tree, the domain sun.com is represented by the entry with the distinguished name o=sun,c=us. This entry is called the *root entry* (sometimes the root domain) of the naming context, and represents the Solaris ISP Server customer's business. You specify the root entry during installation of the directory services.

Beneath the root entry are four required `organizationalUnit` entries:

- *Administrators* contains entries for Sun™ Internet Administrator™ administrators. These entries are created by the product when you create administrators using the GUI.
- *People* contains entries for subscribers to ISP services. You create entries for your customers, whether at the command line or by using the Deja tool.
- *Groups* contains entries that group subscriber entries together for access control purposes. You create group entries according to your needs, whether at the command line or by using the Deja tool.
- *Services* contains entries created for Solaris ISP Server services. You should make entries under this node only when you are integrating a new service.

People, *Groups*, and *Services* nodes are required under each domain entry you define. The *Administrators* node exists only under the root entry.

Figure 3–2 shows a typical set of entries under each organizational unit.

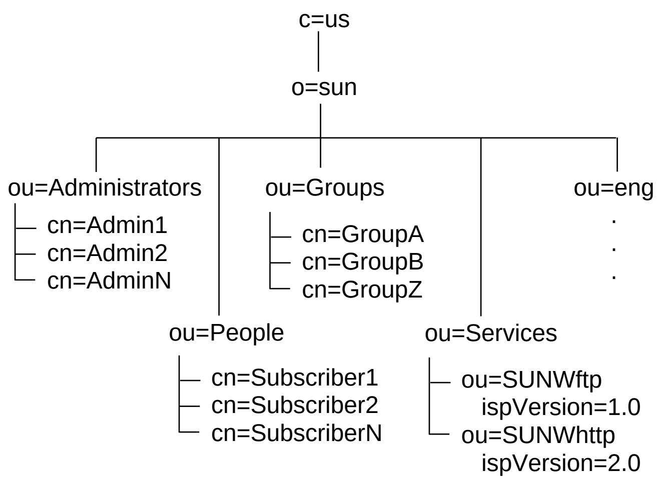
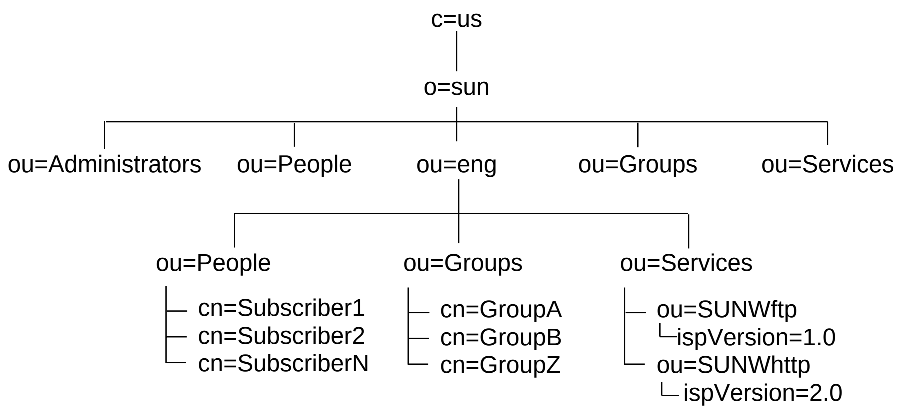


Figure 3–2 OSI Tree Entries

The `organizationalUnit` entry *eng* is an example of a *domain* entry. This might be a corporate customer of the ISP, or anyone who has virtual domain hosting services with the ISP. Domains must have two entries: one here in the OSI tree and another in the DC tree for domain name mapping. See "Creating Domain Entries" on page 39 for information on creating these two entries properly.

Domains, like the root entry, require certain `organizationalUnit` entries within them. As shown in Figure 3–3, *People*, *Groups*, and *Services* entries are also required in a domain below the root.



Note: Data entries in the root domain are omitted for clarity.

Figure 3–3 Domain Structure in the OSI Tree

When creating a domain entry in the OSI tree, you must also create the entries for *People*, *Groups*, and *Services*. When you configure services for this domain, service entries are made under the *Services* organizational unit. Subscriber information for this domain forms `ispSubscriber` entries under the *People* organization unit.

Note - Administrator entries exist only under the root entry in this version of Solaris ISP Server. These entries are created by Sun Internet Administrator when you specify them through the GUI.

DC Tree Structure

The DC (domain component) tree maps domain name format (for example, sun.com) to the distinguished name of the corresponding entry in the OSI tree. As shown in Figure 3–4, the DC tree is usually relatively flat and simpler than the OSI tree.

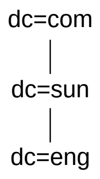


Figure 3–4 Solaris ISP Server DC Tree

In Figure 3–4, the entry dc=sun,dc=com maps to the o=sun,c=us entry in the OSI tree. The *eng* domain here maps to the domain name system (DNS) form eng.sun.com.

For details on how to make the two domain entries, see "Creating Domain Entries" on page 39.

Supporting Multiple Independent Domains

The DC tree in "DC Tree Structure" on page 32 shows an entry for the domain eng.sun.com. Domains that use that same name structure (*.com*) can be added as siblings to *sun*. But ISPs may support many independent top-level domains.. Supporting a new independent domain, such as *.net*, requires adding a new naming context.

To accomplish this, add a new DC tree and make the domain entry under it —in addition to the OSI tree entries and the DNS configuration that are necessary for any domain. To add a new DC tree to the default configuration, follow the steps in "Adding a DC Tree" on page 33.

▼ Adding a DC Tree

You can add a new independent domain to the directory by adding a new DC tree.

1. **Access the Sun Directory Services administration tool, either by launching it from Sun™ Internet Administrator™ or by entering the following:**

```
% su
password:

# /opt/SUNWconn/sbin/dsadmintool
```

2. **In the *Data Store* list, select the data store to which you are adding the naming context.**

   Note - In the default Solaris ISP Server configuration, there is only one data store.

3. **Choose *Modify Data Store* from the *Selected* menu.**

4. **Click *More Suffixes*.**

Note - *More Suffixes* is to the right of the *Data Store Suffix* field. You may have to scroll to see it.

---

5. **In the *Additional Suffix* box, enter the distinguished name of the new naming context. For example, if you are adding a context for *.net*, enter the DN dc=*net*.**

6. **Choose *Naming Context* from the *Create* menu.**

7. **Select a *Type* of *Subtree*.**

8. **Select a *Mode* of *Master*.**

9. **In the *Suffix* field, enter the distinguished name of the new naming context. For example, if you are adding a context for *.net*, enter the DN dc=*net*.**

10. **Click *OK* to save the naming context information. The new naming context is displayed in the *Naming Contexts* section.**

11. **Click *OK* in the *Create Data Store* window to save the data store definition.**

    A dialog appears, asking "Do you really want to apply the modifications?" Click *OK*. If the dialog does not appear, the changes will not take effect and your new DC tree will not be available.

12. **Click *Apply* in the main administration tool window.**

    A dialog appears, asking if you want to restart the directory server and make the new configuration take effect. Click *Yes*. If the dialog does not appear, the changes have not been saved.

13. **Add the root entry for the naming context, using either Deja or `ldapadd(1M)`. For example, if you specified dc=*net*, create an entry with that distinguished name. You cannot add any entries to this data store until this root entry exists.**

Directory Services Access Control

Solaris ISP Server sets access control for the directory services, to ensure proper access by parts of the software that need it while assuring security by preventing access by others. The general principal of these access controls is that most entities have read access while write access is restricted. It is very important that you do not change existing access controls, or you may introduce security risks or cause Solaris ISP Server to fail.

Remember that the access control rules are order sensitive. When Sun Directory Services checks for access, the first rule that applies to the request is used. Any remaining rules are ignored. Therefore, do not change the order of the rules in the file. When creating a new rule, be careful that it does not accidentally apply to existing Solaris ISP Server information and invalidate some access control rule already in place.

Note - Access control checking is switched off if you bind to the directory as its administrator.

Generally, the information special to Solaris ISP Server is stored in entries supporting object classes defined in the Solaris ISP Server schema extension. Each of these classes is named beginning with the string “isp.” Any rule in the access configuration file that contains such an object class (or attribute) is likely a Solaris ISP Server rule and, as such, sensitive to any changes. The access control rules are defined in `/etc/opt/SUNWconn/ldap/current/dsserv.acl.conf`.

For complete information on Sun Directory Services access controls, see Chapter 1, “Introduction to Directory Concepts,” and Chapter 4, “Configuring a Directory Server,” of the *Sun Directory Services 3.1 Administration Guide.*

The sections that follow describe the general behavior ensured by the Solaris ISP Server access controls. The phrase “has access” indicates that binding to the directory with that entry’s DN and password will give the indicated form of access.

Rules Enabling Sun Internet Administrator Functionality

Sun Internet Administrator needs the following kinds of access to do its work:

- It needs to create and delete administrator information. Therefore, Sun Internet Administrator has write access to the portion of the DIT defined by the `Administrators` organizational unit entry.
- It must be able to change certain administrator attributes (notably the `userPassword` and `ispAuthorizedService` attributes).
- It must be able to control the creation of managed service entries (`ispManagedService`). Therefore, Sun Internet Administrator owns its own portion of the tree, below the top-level *SUNWixamc* entry (for example, ispVersion=1.0,ou=SUNWixamc,ou=Services,o=sun,c=us).

- It needs to create the top-level service entries for services it registers and manages. Therefore, Sun Internet Administrator has the access and information it needs to write to that portion of the DIT (for example, ou=Services,o=sun,c=us).
- It also needs to set the value of the protected `ispPrivateData` attribute on `ispService` entries. Therefore, Sun Internet Administrator has read / write access to that attribute of existing service entries. (In fact, no other entity has any access to the `ispPrivateData` attribute.)

Rules Enabling Service Functionality

The various Solaris ISP Server services need to record and access configuration information stored under their service entries (those located under the *Services* node in subdomains and virtual domains). Therefore, each has the access and information it needs to create and modify entries in that portion of the DIT, including its own service entry.

Rules Enabling Proper User Access

Users (subscribers and administrators) have write access to their password attribute, but cannot change other parts of their entry. However, any administrator with management access to Sun Internet Administrator has global access and can change anything.

CHAPTER **4**

Making Directory Entries

This chapter covers creating the domain and NAS (Network Access Server) entries required by Solaris ISP Server™. Both command-line and GUI procedures are included. Topics in this chapter include:

Making Entries from the Command Line

General information on how to create directory services entries is located in Chapter 5, "Loading and Maintaining Directory Information," in the *Sun Directory Services 3.1 Administration Guide*. This section presents instructions on how to create the directory entries required by Solaris ISP Server.

Creating Directory Entries: General Procedure

Sun Directory Services has the following command-line utilities for creating and modifying directory entries:

- `ldapadd`
- `ldapmodify`
- `ldapdelete`

These directory services command-line utilities require root access. They are fully documented in reference manual pages (section 1).

Both `ldapadd` and `ldapmodify` can take input from the command line or from a specified file. Because information for an entry can be rather lengthy and complex, the sections that follow describe the form requiring a text file.

In each case, creating an entry (or entries) requires the following steps:

1. **Write a file specifying the entry or entries to be made in the directory. The format for this file is specified in the `ldif(4)` reference manual page.**

2. **Create the entry using `ldapadd`, specifying the file with the entry information.**

In every case, the form of the `ldapadd` command should be:

`ldapadd –D` *"BindDN"* `–w` *password* `–f` *file*

Where *BindDN* is the distinguished name (DN) for binding to the directory with write access to this part of the directory tree, and *password* is the password for binding. Replace the *file* option with the name of the `ldif` file you have created.

Structure of an `ldif(4)` File

For each entry you add at the command line, you will create an entry in an `ldif`-format file to hold the information about the entry. These are simple text files with one or more directory entries each separated by a single blank line. Each entry has the structure of the following example.

Note - Only mandatory attributes are shown in the example. Most object classes have a number of optional attributes that may be set appropriately for your particular use of the entry.

The example shows an entry for a record with the `organizationalUnit` object class. Other records with different object classes will have different attributes. The example is to show the format.

```
dn: ou=wcgate1,ou=eng,o=sun,c=US
ou: wcgate1
associateddomain: wcgate1.eng.sun.com
objectclass: organizationalUnit
objectclass: domainRelatedObject
```

Where

dn	Indicates the distinguished name of the entry being created. Use the DN for your desired domain.
ou	Is the naming attribute of the entry being created. Common naming attributes include `commonName`, `organizationalUnit` (`ou`), and `domainComponent` (`dc`). Use the RDN for your domain.
associatedDomain	Contains the domain name (in dot notation) of the corresponding entry in the DC tree. Use the name of your domain. See “Solaris ISP Server Directory Structure” on page 30 for information on how the OSI tree and the DC tree interact. See “Creating Domain Entries” on page 39 for instructions on creating the two cross-referenced entries for a domain. There may be many attribute:value pairs in this position, one per line.
objectClass	Is the object class (type) of the entry. There may be many `objectClass` entries; this example shows two.

For more detailed information on available object classes and attributes, see Chapter 6 of this guide, and Chapter 8, “Configuring the Directory Schema,” of the *Sun Directory Services 3.1 Administration Guide.*

Creating Domain Entries

To create a domain in the directory, you must create two parallel domain entries, one in the OSI tree and one in the DC tree, and then create the required `organizationalUnit` entries under the domain entry in the OSI tree.

To create the domain *wcgate1* under eng.sun.com, perform the following steps:

1. **Edit a text file (for example, `domain.ldif`) and enter the data for the OSI tree entry:**

```
dn: ou=wcgate1,ou=eng,o=sun,c=US
ou: wcgate1
associateddomain: wcgate1.eng.sun.com
objectclass: organizationalUnit
objectclass: domainRelatedObject
```

Note that the `associatedDomain` attribute of the entry contains the DNS name of the domain.

2. **Add to** `domain.ldif` **the data for the DC tree entry:**

```
dn: dc=wcgate1,dc=eng,dc=sun,dc=com
dc: wcgate1
associatedname: ou=wcgate1,ou=eng,o=sun,c=US
description: DNS-to-DN Mapping for wcgate1.eng.sun.com
labeleduri: ldap:///ou=wcgate1,ou=eng,o=sun,c=US??sub
objectclass: domain
objectclass: labeledURIObject
```

Note that the `associatedName` attribute of the entry contains the distinguished name of the OSI tree entry. The `labeledURI` attribute contains the same information (as specified in RFC 2255).

3. **Add to** `domain.ldif` **the data for the required** `Services` **organizational unit entry:**

```
dn: ou=Services,ou=wcgate1,ou=eng,o=sun,c=US
ou: Services
objectclass: organizationalUnit
```

4. **Add to** `domain.ldif` **the data for the required** `People` **organizational unit entry:**

```
dn: ou=People,ou=wcgate1,ou=eng,o=sun,c=US
ou: People
objectclass: organizationalUnit
```

5. **Add to** `domain.ldif` **the data for the required** `Groups` **organizational unit entry:**

```
dn: ou=Groups,ou=wcgate1,ou=eng,o=sun,c=US
ou: Groups
objectclass: organizationalUnit
```

6. **Save and close** `domain.ldif`**.**

7. **Add the entries to the directory with the following command, replacing the bind DN and password with your own:**

```
% ldapadd -D "cn=admin,o=sun,c=US" -w secret -f domain.ldif
```

When your `ldapadd` is complete, the directory looks like Figure 4–1.

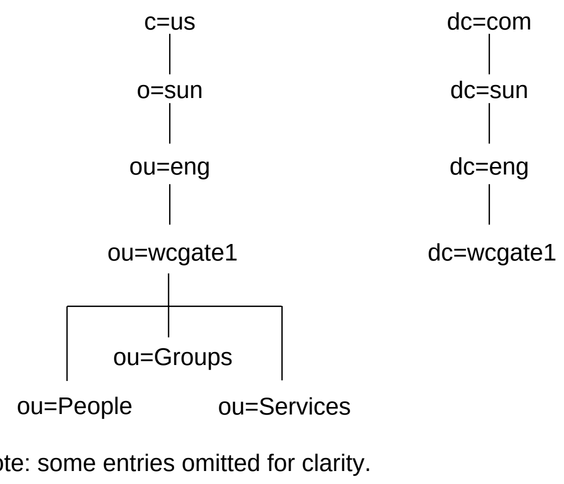


Figure 4–1 Directory Structure with a Domain Added

Creating Group Entries

Before you can create group entries, a number of entries must already exist:

- The two domain entries (OSI and DC trees)
- The *Group* organizational unit entry
- The subscriber entries (under the *People* node) that will become the members of the group.

Once you have created those entries, you can start a text file (for example, `groups.ldif`) and enter the data for the group. A typical data set looks like the following:

dn: cn=*isp-gp1*,ou=Groups,ou=*wcgate1*,ou=*eng*,o=*sun*,c=*US*
cn: *isp-grp1*
objectclass: groupOfNames
member: cn=*Ed Anchor (anchor)*,ou=People,ou=*wcgate1*,ou=*eng*,o=*sun*,c=*US*
member: cn=*April Shower (showers)*,ou=People,ou=*wcgate1*,ou=*eng*,o=*sun*,c=*US* S
member: cn=*Chili Jones (relleno)*,ou=People,ou=*wcgate1*,ou=*eng*,o=*sun*,c=*US*

Where

dn	Is the distinguished name of the group to be created. Use a group name appropriate to your environment.
cn	Is the relative distinguished name of the group entry.
objectClass	The object class `groupOfNames` distinguishes this type of entry.
member	Each `member` attribute takes as its value the distinguished name of an existing subscriber entry.

You can create any number of group entries by adding data to the file. When it is complete, save and close `groups.ldif`. Add the groups to the directory with the following command, replacing the bind DN and password with your own:

```
% ldapadd -D "cn=admin,o=sun,c=US" -w secret -f groups.ldif
```

Making Entries Using Deja

▼ Accessing the Sun Directory Services Deja Tool

You can create individual entries in the directory services with the Deja tool. To access Deja:

1. **If working remotely, log into the machine where the directory services server is running. Set the environment variable DISPLAY to allow the browser-based GUI to work remotely.**

2. **To run Deja as a Java application, enter:**

   ```
   /opt/SUNWconn/bin/deja
   ```

3. **Log into Deja, using the Sun Directory Services administrator's distinguished name and password. The successful login message displays in Deja's message box (lower left). You can proceed to perform any tasks requiring Deja.**

Creating Directory Entries for a Domain

Five directory entries are necessary to support a domain: one in the DC tree and four in the OSI tree (see Chapter 3 for illustrations of the tree structure). Follow the steps in "Creating the DC Tree Entry" on page 43, "Creating the OSI Tree Entry" on page 44, and "Creating the `organizationalUnit` Entries" on page 45 to complete the required domain entries.

Note - You must also perform any DNS or NIS mapping your domains require.

▼ Creating the DC Tree Entry

The DC tree entry maps the domain name server form of the name to the distinguished name of the entry in the OSI tree. Be very careful to enter the `associatedName` attribute correctly; it must contain the distinguished name of the OSI tree entry.

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the domain's parent in the parent text field. For example, for a domain beneath *sun.com* you would enter: dc=*sun*,dc=*com*.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

4. **Choose the** `dc` **attribute for the domain's relative distinguished name, and enter the name. Click *Next Step*.**

5. **Choose the object class** `domain` **and click *Next Step*.**

6. **Set values for the following mandatory attributes:**
   - `dc`: enter the relative distinguished name of the domain.

   Note - It may appear that the `dc` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `dc` attribute. Enter the same value.

   - `associatedName`: enter the domain name of the domain entry you will create in the OSI tree. For example, the domain *eng* beneath *sun.com* would have the distinguished name ou=*eng*,o=*sun*,c=*us*.

7. **Click *Done*.**

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

8. **Proceed to the next section and create the OSI tree entry.**

▼ Creating the OSI Tree Entry

The OSI tree entry contains the actual directory services information about the domain. Be very careful to enter the `associatedDomain` attribute correctly; it must contain the name of the DC tree entry in domain name form, for example *eng.sun.com*.

1. **Click *Cancel* to get an empty *Create Entry* screen in Deja.**

2. **Enter the distinguished name of the domain's parent in the parent text field. For example, for a domain beneath *sun.com* you would enter: o=*sun*,c=*us*.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

3. **Choose the** `ou` **attribute for the domain's relative name, and enter the name. The** `ou` **attribute is case-insensitive.**

4. **Click *Next Step*.**

5. **Choose the object classes** `organizationalUnit` **and** `domainRelatedObject` **and click *Next Step*.**

6. **Set values for the following mandatory attributes:**
   - `ou`: enter the relative name of the domain. The `ou` attribute is case-insensitive.

     Note - It may appear that the `ou` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `ou` attribute. Enter the same value.

   - `associatedDomain`: enter the distinguished name of the domain entry you just created. For example, *eng.sun.com*.

7. **Click *Done*.**

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

8. **Proceed to the next section and create the** `organizationalUnit` **entries.**

▼ Creating the `organizationalUnit` Entries

Solaris ISP Server expects subscriber, group, and service entries to be located in specific `organizationalUnit` entries under a domain or organization. Once your domain entries are correct, create these entries as well.

1. **Click *Cancel* to get an empty *Create Entry* screen in Deja.**

2. **Enter the distinguished name of the domain you just created in the parent text field. For example, if your domain were *eng.sun.com*, you would enter ou=*eng*,o=*sun*,c=*us*.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

3. **Choose the** `ou` **attribute for the** `organizationalUnit's` **relative distinguished name, and enter *People*. The** `ou` **attribute is case-insensitive.**

4. **Click *Next Step*.**

5. **Choose the object class** `organizationalUnit` **and click *Next Step*.**

6. **Set the value of the** `ou` **attribute to *People*.**

7. **Click *Done*.**

8. **Repeat steps 1 through 7 for the *Groups* and *Services* organization unit entries.**

▼ Creating a NAS Entry for the RADIUS Server

The network access server (NAS) is the server running within a terminal server. It contacts the Sun Directory Services RADIUS server for authentication of the user requesting access.

To make a NAS entry in the directory:

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber's parent in the parent text field; for example, ou=*Services*,ou=*SomeDomain*,o=*sun*,c=*us*.**

Note - The default Solaris ISP Server configuration assumes that NAS entries are located under the Services node in the root domain. If you locate yours elsewhere, see the RADIUS mapping manual page on how to modify the `/etc/opt/SUNWconn/ldap/current/mapping/radius.mapping` configuration file to reflect your environment.

---

4. **Choose the `cn` attribute for the NAS entry's relative distinguished name and enter the name of the network access server. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object classes `nas` and `device`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `cn`: enter the name of the network access server. The `cn` attribute is case-insensitive.

   Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

   - `sharedKey`: enter the key shared with your network access server.

- `ipHostNumber`: enter the IP address of the host where the network access server is running.

8. **Click *Done*.**

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

▼ Creating Group Entries

Sun WebServer uses groups to control access to material. When a realm is configured to use the directory services as its source, the users and groups must be entries in Sun Directory Services. In order to create a group entry, the individual members of the group must already have `ispSubscriber` entries in the directory.

To make a group entry in the directory:

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber's parent in the parent text field; for example, ou=*Groups*,ou=*SomeDomain*,o=*sun*,c=*us*. Solaris ISP Server expects subscriber entries to be located under a *Groups* node in a domain or organization.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

4. **Choose the `cn` attribute for the group's relative distinguished name, and enter the name of the group. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object class `groupOfNames`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `cn`: enter the name of the group. The `cn` attribute is case-insensitive.

Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

- `member`: enter the distinguished name of the first member and click *Add*. Assign one `member` attribute and value for each actual member of the group. Each entry is the distinguished name of an `ispSubscriber` entry.

8. Click *Done.*

The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

CHAPTER 5

Making Subscriber Directory Entries

This chapter covers creating subscriber entries in the Solaris ISP Server™ directory service using command line tools, the Deja tool, or by converting existing user data in bulk. Topics include:

Making Subscriber Entries from the Command Line

Solaris ISP Server subscribers come in several varieties:

- The general (basic) subscriber
- The subscriber who uses virtually-hosted FTP or Web services
- The subscriber who gains access to services through a RADIUS server
- The subscriber who uses both, and whose directory entry requires both RADIUS and FTP information

In the sections that follow, instructions are provided for building the complex subscriber entry by creating the simpler entry and adding to it.

Creating a Basic Subscriber Entry

Before you can create subscriber entries, the domain and the *People* organizational unit entries must exist. Once you have created those entries, you can edit a text file (for example, `people.ldif`) and enter the data for the subscriber. The basic subscriber entry has the single object class `ispSubscriber`, and very few mandatory attributes. The file for a basic subscriber looks like this:

dn: cn=*Jane Doe (jldoe)*,ou=People,ou=*wcgate1*,ou=*eng*,o=*sun*,c=*US*
commonname:*Jane Doe (jldoe)*
sn: *Doe*
uid: *jldoe*
userpassword: *hidden*
objectclass: ispSubscriber

Where

dn	Is the distinguished name of the subscriber entry. Use the appropriate DN for subscriber entries in the desired domain.
commonName	Is the naming attribute of a subscriber entry (`ispSubscriber` object class). For Solaris ISP Server subscribers and administrators, the value of the `commonName` attribute takes the form **`Firstname Lastname (userid)`**.
sn	Is the surname of the subscriber.
uid	Is the login name of the subscriber.
userPassword	Is the password, limited to eight characters if you are sharing password information with UNIX accounts. This value is generated with the encryption method you set in the directory services administration console.

objectClass: ispSubscriber Is the object class type of this subscriber entry.

You can create any number of subscriber entries by adding blocks of data with different attribute values to the file. When it is complete, save and close `people.ldif`. Add the subscriber entries to the directory with the following command, replacing the bind DN and password with your own:

```
% ldapadd -D "cn=admin,o=sun,c=US" -w secret -f people.ldif
```

Adding FTP and Web Virtual Hosting Information

The information required for the specially-configured virtual hosting available with Sun™ Internet FTP Server™ and Sun™ WebServer™ adds only three attributes to the data file:

gidnumber: *60001*
uidnumber:*60001*
ispcontentdirectory: *jldoe*

Where

gidNumber	Is the UNIX group ID specified for this user in the virtually-hosted domain for FTP and Web services. Use the GID appropriate to your use of UNIX accounts.
uidNumber	Is the UNIX user ID specified for this user in the virtually-hosted domain for FTP and Web services. Use the UID appropriate to your use of UNIX accounts.
ispContentDirectory	Is the location (relative to the associated domain's document root) where this subscriber's content files are located.

Note - Setting the values for the `uidNumber` and `gidNumber` attributes requires existing UNIX accounts properly set up to share access to the virtual FTP domain. See the Sun Internet FTP Server online help for information on defining a virtual host configuration.

You can create any number of subscriber entries by adding blocks of data to the file. When it is complete, save and close `people.ldif`. Add the subscriber entries to the directory with the following command, replacing the bind DN and password with your own:

```
% ldapadd -D "cn=admin,o=sun,c=US" -w secret -f people.ldif
```

If you have already created these entries, you must perform an `ldapmodify`. Locate the manual page for `ldapmodify(1)` and follow those instructions.

Adding Remote User Information

An entry for a subscriber who gains access to ISP services through a RADIUS server must support an additional object class (`remoteUser`) and has several attributes added to the entry information.

Note - The default Solaris ISP Server configuration designates the root domain as the search base for RADIUS subscriber entries. If your configuration is different, use the directory services administration console to configure RADIUS and enter values appropriate for your search base.

The additional lines in the `ldif` file are:

objectclass: remoteUser
authsuffixname: *@ispxpress*
grpcheckinfo: authSuffixName
grpcheckinfo: userPassword
authserviceprotocol: *Framed-User*
framedrouting: *None*
framedprotocol: *PPP*
grpreplyinfo: authServiceProtocol
grpreplyinfo: framedProtocol
grpreplyinfo: framedRouting

Where

objectClass: remoteUser	Is a required object class for the subscriber accessing services using a RADIUS server.
authsuffixname: @ispxpress	Is a suffix added to the subscriber's user name to enable the RADIUS server to distinguish among entries with the same `uid` in different domains. Use the appropriate suffix for the your user entry.
grpcheckinfo: authSuffixName	Indicates that the RADIUS server should verify the `authSuffixName` attribute

	value before selecting the entry to authenticate against.
grpcheckinfo: userPassword	Indicates that the RADIUS server should verify the `userPassword` attribute value before selecting the entry to authenticate against.
authserviceprotocol: Framed-User	If you are using the default RADIUS configuration, enter this attribute exactly as shown. The correct value is determined by the configuration of your network access server.
framedrouting: None	If you are using the default RADIUS configuration, enter this attribute exactly as shown. The correct value is determined by the configuration of your network access server.
framedprotocol: PPP	If you are using the default RADIUS configuration, enter this attribute exactly as shown. The correct value is determined by the configuration of your network access server.
grpreplyinfo: authServiceProtocol	Tells the RADIUS server to include the value of the `authServiceProtocol` attribute in its reply message.
grpreplyinfo: framedProtocol	Tells the RADIUS server to include the value of the `framedProtocol` attribute in its reply message.
grpreplyinfo: framedRouting	Tells the RADIUS server to include the value of the `framedRouting` attribute in its reply message.

You can create any number of subscriber entries by adding blocks of data to the file. When it is complete, save and close `people.ldif`. Add the subscriber entries to the directory with the following command, replacing the bind DN and password with your own:

```
% ldapadd -D "cn=admin,o=sun,c=US" -w secret -f people.ldif
```

If you have already created these entries, you must perform an `ldapmodify`. Locate the manual page for `ldapmodify(1)` and follow those instructions.

The Complete `ldif` File

The complete `ldif` file for a complex user looks like:

```
dn: cn=Jane Doe (jldoe),ou=People,ou=wcgate1,ou=eng,o=sun,c=US
commonname:Jane Doe (jldoe)
sn: Doe
uid: jldoe
userpassword: hidden
gidnumber: 60001
uidnumber: 60001
objectclass: ispSubscriber
objectclass: remoteUser
ispcontentdirectory: /home/users/jldoe
authsuffixname: @ispxpress
grpcheckinfo: authSuffixName
grpcheckinfo: userPassword
authserviceprotocol: Framed-User
framedrouting: None
framedprotocol: PPP
grpreplyinfo: authServiceProtocol
grpreplyinfo: framedProtocol
grpreplyinfo: framedRouting
```

Making Subscriber Entries Using the Deja Tool

▼ Creating a Basic Subscriber Entry

Each ISP subscriber (or customer) must have an entry in the directory. These entries, of the `ispSubscriber` object class, provide authentication and access information for each customer. There are different types of subscriber entries, depending on the needs of the individual subscriber.

To make a basic subscriber entry in the directory:

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber's parent in the parent text field; for example, ou=People,ou=SomeDomain,o=sun,c=us. Solaris ISP Server expects subscriber entries to be located under a *People* node in a domain or organization.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

4. **Choose the `cn` attribute for the subscriber's relative distinguished name, and enter the name in the form `Firstname Lastname (userid)`. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object class `ispSubscriber`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `userid`: enter the user's login name
   - `commonName`: enter the subscriber's name, in the form **`Firstname Lastname (userid)`**. The `cn` attribute is case-insensitive.

     Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

   - `sn`: enter the subscriber's surname.
   - `userPassword`: enter the user's password. Limit passwords to 8 characters if you are sharing user information with UNIX accounts.

   These attributes must have values for the subscriber entry to be valid. You can add values for optional entries as appropriate.

8. **Click *Done*.**

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

   See the Sun Directory Services User Guide, Chapter 2, "Deja Tool: Standard LDAP Features" for more information about the Deja tool.

▼ Creating an Entry for a Subscriber with Virtual Hosting

To make a subscriber entry in the directory:

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber's parent in the parent text field; for example, ou=People,ou=SomeDomain,o=sun,c=us. Solaris ISP Server expects subscriber entries to be located under a *People* node in a domain or organization.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

4. **Choose the `cn` attribute for the subscriber's relative distinguished name, and enter the name in the form `Firstname Lastname (userid)`. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object class `ispSubscriber`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `userid`: enter the user's login name
   - `commonName`: enter the subscriber's name, in the form **`Firstname Lastname (userid)`**. The `cn` attribute is case-insensitive.

   Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

   - `sn`: enter the subscriber's surname.
   - `userPassword`: enter the user's password. Limit passwords to 8 characters if you are sharing user information with UNIX accounts.
   - `gidNumber`: enter the UNIX group ID defined for a user in the virtually-hosted domain.

- `uidNumber`: enter the UNIX user ID defined for a user in the virtually-hosted domain.
- `ispContentDirectory`: enter the path (relative to the `ispDirectoryRoot` defined for the virtually-hosted domain) to the subscriber's FTP content directory. Include the leading slash, for example: /home/ftp.

These attributes must have values for the subscriber entry to be valid. You can add values for optional entries as appropriate.

8. Click *Done.*

The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

See the Sun Directory Services User Guide, Chapter 2, "Deja Tool: Standard LDAP Features" for more information about the Deja tool. Also see Appendix A, "Configuring a Virtual Host."

▼ Creating an Entry for a Subscriber with RADIUS Access

Adding RADIUS server access information to a subscriber entry involves adding an object class, `remoteUser`, and setting a number of special attributes. To make a subscriber entry in the directory for a user who accesses services through a RADIUS server:

1. **Start and log into Deja. See "Accessing the Sun Directory Services Deja Tool " on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber's parent in the parent text field; for example, ou=People,ou=SomeDomain,o=sun,c=us. Solaris ISP Server expects subscriber entries to be located under a *People* node in a domain or organization.**

Tip - If the parent is visible in the tree on the left of Deja's screen, select it and click *Get from Browser*.

---

4. **Choose the `cn` attribute for the subscriber's relative distinguished name, and enter the name in the form `Firstname Lastname (userid)`. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object classes `ispSubscriber` and `remoteUser`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `userid`: enter the user's login name
   - `commonName`: enter the subscriber's name, in the form **`Firstname Lastname (userid)`**. The `cn` attribute is case-insensitive.

     Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry's relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

   - `sn`: enter the subscriber's surname.
   - `userPassword`: enter the user's password. Limit passwords to 8 characters if you are sharing user information with UNIX accounts.
   - `gidNumber`: enter the UNIX group ID defined for a user in the virtually-hosted domain.
   - `uidNumber`: enter the UNIX user ID defined for a user in the virtually-hosted domain.
   - `ispContentDirectory`: enter the path (relative to the `ispDirectoryRoot` defined for the virtually-hosted domain) to the subscriber's FTP content directory. Include the leading slash, for example: /home/ftp.
   - `authSuffixName`: enter the suffix string to be appended to the user's login name.
   - `grpCheckInfo`: enter the attributes you want used for user authentication (`grpCheckInfo` is multi-valued). We recommend that you enter at least `authSuffixName` and `userPassword`.
   - `authServiceProtocol`: enter `Framed-User`.
   - `framedRouting`: enter `None`.
   - `framedProtocol`: enter `PPP`.
   - `grpReplyInfo`: set this multivalued attribute three times, with the following values:
     - **`authServiceProtocol`**
     - **`framedProtocol`**
     - **`framedRouting`**

   These attributes must have values for the subscriber entry to be valid. You can add values for optional entries as appropriate.

8. **Click *Done*.**

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

See the Sun Directory Services User Guide, Chapter 2, "Deja Tool: Standard LDAP Features" for more information about the Deja tool. Also see Appendix A, “Configuring a Virtual Host.”

▼ Creating an Entry for a Subscriber with both Virtual Hosting and RADIUS Access

To make an entry in the directory services for a subscriber who uses a RADIUS server for access and who has FTP or Web services on a virtual host:

1. **Start and log into Deja. See “Accessing the Sun Directory Services Deja Tool ” on page 42 for detailed steps.**

2. **Choose *Create Entry* from the *Entry* menu.**

3. **Enter the distinguished name of the subscriber’s parent in the parent text field; for example, ou=People,ou=SomeDomain,o=sun,c=us. Solaris ISP Server expects subscriber entries to be located under a *People* node in a domain or organization.**

Tip - If the parent is visible in the tree on the left of Deja’s screen, select it and click *Get from Browser*.

---

4. **Choose the `cn` attribute for the subscriber’s relative distinguished name, and enter the name in the form `Firstname Lastname (userid)`. The `cn` attribute is case-insensitive.**

5. **Click *Next Step*.**

6. **Choose the object classes `ispSubscriber` and `remoteUser`, and click *Next Step*.**

7. **Set values for the following mandatory attributes:**
   - `userid`: enter the user’s login name
   - `commonName`: enter the subscriber’s name, in the form **`Firstname Lastname (userid)`**. The `cn` attribute is case-insensitive.

   Note - It may appear that the `cn` attribute must be set twice. The first step sets the entry’s relative distinguished name. This step actually sets the `commonName` attribute. Enter the same value.

- `sn`: enter the subscriber's surname.
- `userPassword`: enter the user's password. Limit passwords to 8 characters if you are sharing user information with UNIX accounts.
- `authSuffixName`: enter the suffix string to be appended to the user's login name.
- `grpCheckInfo`: enter the attributes you want used for user authentication (`grpCheckInfo` is multi-valued). We recommend that you enter at least `authSuffixName` and `userPassword`.
- `authServiceProtocol`: enter `Framed-User`.
- `framedRouting`: enter `None`.
- `framedProtocol`: enter `PPP`.
- `grpReplyInfo`: set this multivalued attribute three times, with the following values:
  - **`authServiceProtocol`**
  - **`framedProtocol`**
  - **`framedRouting`**

  These attributes must have values for the subscriber entry to be valid. You can add values for optional entries as appropriate.

8. **Click *Done.***

   The message "Entry successfully created" appears in the Deja message box. Your entry appears in the directory tree graph on the left of the Deja screen.

   See the Sun Directory Services User Guide, Chapter 2, "Deja Tool: Standard LDAP Features" for more information about the Deja tool. Also see Appendix A, "Configuring a Virtual Host."

Bulk Loading Subscriber Entries

The Solaris ISP Server Directory Services component contains tools for converting existing user data into entries in the directory service. You can export existing data, for example, from a database, NIS, or another LDAP directory into a structured text file, then use the `sispload` tool to convert the data into an ldif file that can be used with `ldapmodify`.

The file `/etc/opt/SUNWconn/ldap/current/mapping/sispload.mapping` contains tables for mapping input data into one of four types of user entry:

- The `sisp` table creates `ispSubscriber` entries.

- The `sisprad` table creates `ispSubscriber` entries with the `remoteUser` object class for RADIUS authentication.
- The `sispsims` table creates `ispSubscriber` entries with the `emailPerson` object class for use with Sun Internet Mail Server software.
- The `sispradsims` table creates `ispSubscriber` entries with the `emailPerson` and `remoteUser` object classes for use with Sun Internet Mail Server software and RADIUS authentication.

For full details on the mapping tables, see the **`sispload.mapping`**(4) man page in `/opt/SUNWisp/man`.

▼ Making Bulk Entries from Existing User Data

1. **Export your existing user data to a text file (the "input file") with one user on each line.**

   You may need to edit the text file to get the data in the correct order and to use the proper field separator. If possible, arrange the data in the order described in Step 2 on page 61 to reduce the amount of manual editing required.

   An example input file is provided in `/opt/SUNWisp/ldap/sunds/examples/sispload.input`.

2. **Edit the input file to use the proper field separator and to arrange the fields in the order expected by `sispload`.**

   By default, fields must be separated by an exclamation point (!). To change the field separator, see the **`sispload.mapping`**(4) manual page.

   The order of the fields for the `sisp` table is described below. If you wish to use the `sisprad`, `sispsims`, or `sispradsims` tables, consult the **`sispload.mapping`**(4) manual page.

   Each line should have the following fields in this order (optional fields with no data may be left blank, but a field separator is still required; see the **`sispload.mapping`**(4) manual page for details):

   ```
   !LastName!FirstName!Nickname!Username!Password!DomainName!\
        E-mailAddress!Services!UID!GID!ContentDirectory!
   ```

   The `Services` and `ContentDirectory` fields may not exist in your current user database. Creating data for these fields is explained in Step 3 on page 62.

   Here is a sample input line with the `Password` field left blank (the `Username` will be entered as the password by `sispload`):

   ```
   !Smith!Joseph!Joe!jsmith!!myIsp.net!jsmith@pop.myIsp.net!\
   ispVersion=1.0,ou=SUNWftp,ou=Services,o=myIsp,c=us!\
   90981!100!jsmith/public_html!
   ```

3. **Since the** `Services` **and** `ContentDirectory` **fields may not exist in your existing data, you may need to edit the input file to add these fields.**

   The `Services` field should contain the distinguished name of each `ispService` entry representing a service the user is allowed to access (for example, FTP or news). For each service, the distinguished name must include the `ispVersion` attribute, an `ou` entry with the service ID, and a relative distinguished name from the root of the tree to an `ou=Services` node. For example:

   ```
   ispVersion=1.0,ou=SUNWftp,ou=Services,o=myIsp,c=us
   ```

   The `ContentDirectory` field specifies a directory where the user may store personal files such as web pages. The `sispload.mapping` file specifies "/home" as the default root, and the value in the `ContentDirectory` field is appended to create the value of the `ispContentDirectory` attribute for the `ispSubscriber`. Note that the `ispContentDirectory` is relative to the `ispDirectoryRoot` of an `ispService` entry that makes use of it. Edit `sispload.mapping` to use a different default root.

4. **Run** `/opt/SUNWisp/ldap/sunds/sbin/sispload` **with the input file to create an ldif file.**

   The `sispload` command takes the following arguments:

   `sispload` [`–b` *baseDN*] [`–d` *debugLevel*] [`–h` *ldapHost*] [`–m` *mapFile*]
   [`–o` *outputFile*] [`–t` *mapTable*] [`–V` *variable*=*value*`...`]`–f` *inputFile*

`–b` *baseDN*	Specifies the base distinguished name beneath which the user entries will be added. This argument is optional, and defaults to the primary naming context you specify at install time. You may want to specify a different domain in your directory tree (for example, "`ou=subDomain,o=myIsp,c=US`") .
`–d` *debugLevel*	Specifies the level of logging performed by the ldap server daemon (`dsservd`). See the manual page for **`sispload`**(1M) for values.
`–h` *ldapHost*	Specifies the host name of the Solaris ISP Server directory server. By default, the host from which `sispload` is invoked is used.
`–m` *mapFile*	Specifies the mapping file that contains the definitions for translating the input file into ldif entries. By default `sispload.mapping` in the directory `/etc/opt/SUNWconn/ldap/current/mapping` is used.
`–o` *outputFile*	Specifies a prefix for creating files with the error and ldif output. By default, errors are sent to `STDERR` and the ldif

	output to STDOUT. If you specify a value here, two files will be created: *outputFile*`.ldif` and *outputFile*`.err`.
`–t` *mapTable*	Specifies the table in the mapping file to use. By default, the `sisp` table in the default mapping file is used; other options in the default mapping table are `sisprad`, `sispsims`, and `sispradsims`.
`–V` *variable*=*value*	Sets variables for use in the conversion. Repeat the `–V` flag to define multiple variables.
`–f` *inputFile*	Specifies the full path name to the input file you have created. This option is required.

For example:

```
# cd /opt/SUNWisp/ldap/sbin
# ./sispload -b o=myIsp,c=US -t sisprad -o /tmp/ldapusers -f /tmp/input.nis.txt
```

5. **Run** `/opt/SUNWconn/bin/ldapmodify` **with the ldif file you created to add the users to the directory.**

   For example:

```
# cd /opt/SUNWconn/bin
# ./ldapmodify -D cn=admin,o=myIsp,c=US -w secret -f /tmp/ldapusers.ldif
```

6. **Examine the error output to see if any entries could not be added to the directory service.**

   The error output will contain ldif entries for users that could not be added to the directory. Usually the cause is a syntax error in the input file. You may want to extract these entries from the original input file, save them to a new file, fix the errors, and run `sispload` again.

CHAPTER **6**

Solaris ISP Server Directory Schema

Solaris ISP Server™ 2.0 extends the standard schema defined in Sun™ Directory Services. The base schema is defined in Chapter 8, "Configuring the Directory Schema," of the *Sun Directory Services 3.1 Administration Guide*. Extensions to the base schema, both object classes and attributes, are discussed in this chapter.

Note - The schema defined by Solaris ISP Server is unstable in this release of the product. Object class and attribute definitions that are a part of the schema extension may change without warning in future releases. See "Avoiding Schema Dependence" on page 68 for guidelines on avoiding schema dependence in your code.

This chapter includes the following topics:

- "Maintaining the Schema" on page 65
- "Avoiding Schema Dependence" on page 68
- "Solaris ISP Server Object Classes" on page 69
- "Solaris ISP Server Attributes" on page 76
- "Schema Changes for Sun™ Internet Mail Server" on page 83

Maintaining the Schema

In general, you should not change existing object classes or attributes in the schema, but use or add to them for your purposes. If you change an attribute or object class so that the Solaris ISP Server software cannot use it, you may have to reinstall the directory. In this case, any data entries not backed up will be lost. For information on backing up the directory, see the Sun Directory Services manual pages for `ldbmcat(1M)` and `ldif2ldbm(1M)`.

If you decide to add to the schema, refer to Chapter 8, “Configuring the Directory Schema,” of the *Sun Directory Services 3.1 Administration Guide* for complete details.

What to Back Up

To ensure the integrity of the directory, you should periodically back up the schema configuration files. Certainly, you should back up the schema before starting work to extend it for your own uses.

As discussed in detail in Chapter 4, “Configuring Directory Services,” of the *Sun Directory Services 3.1 Administration Guide*, the following files are critical for directory services function:

- `dsserv.conf` holds the main configuration information.
- `dsserv.oc.conf` holds the object class definitions.
- `dsserv.at.conf` holds the attribute definitions.
- `dsserv.acl.conf` holds access control information.
- `SUNWisp.dsserv.at.conf` holds the attribute definitions for Solaris ISP Server schema extensions.
- `SUNWisp.dsserv.oc.conf` holds the object class definitions for Solaris ISP Server schema extensions.

Copies of these files are stored in three places in the system:

- `/etc/opt/SUNWconn/ldap/current` holds the current configuration files.
- `/etc/opt/SUNWconn/ldap/default` holds the default configuration files that were installed with the software. (These files are read-only.)
- `/etc/opt/SUNWconn/ldap/previous` holds the previous configuration files.

Before starting any work on the schema, back up the configuration files in `/etc/opt/SUNWconn/ldap/current` and those in `/etc/opt/SUNWconn/ldap/previous`. When you edit the files through dsadmintool or restart the daemon, Sun Directory Services copies the unedited files in `/etc/opt/SUNWconn/ldap/current` to `/etc/opt/SUNWconn/ldap/previous`. It does this only once per editing session, (until you restart `dsservd`). If you are making many changes to the schema, you may want to make manual backups of your changes as you work.

Backup information is presented in detail in Chapter 4, “Configuring Directory Services,” of the *Sun Directory Services 3.1 Administration Guide*.

Restoring the Schema

To restore your directory services configuration to a previous version, stop `dsservd`, replace the desired configuration files in `/etc/opt/SUNWconn/ldap/current`, and restart the daemon. Step-by-step information is presented in Chapter 4, “Configuring Directory Services,” of the *Sun Directory Services 3.1 Administration Guide*.

Restoring the Solaris ISP Server Schema

When Solaris ISP Server is installed, the original schema configuration files are backed up at `/etc/opt/SUNWisp/ldap/sunds/default`. These files were customized at installation with your root domain and administrator information. If you are working on the schema, and arrive at a situation where Solaris ISP Server does not work, restore the default schema as follows:

1. **Log into the machine where the directory services is running.**

2. **Give yourself root access.**

3. **Stop the directory services server, by entering:**
   ```
   # /etc/init.d/dsserv stop
   ```

   Note - You can also stop and start the server through its administration console.

   ---

4. **Copy each of the configuration files from the default to the current directory:**

```
% cp /etc/opt/SUNWisp/ldap/sunds/default/dsserv.conf \
/etc/opt/SUNWconn/ldap/current/dsserv.conf

% cp /etc/opt/SUNWisp/ldap/sunds/default/SUNWisp.dsserv.at.conf \ /etc/opt/SUNWconn/ldap/current/SUNWisp

% cp /etc/opt/SUNWisp/ldap/sunds/default/SUNWisp.dsserv.oc.conf \ /etc/opt/SUNWconn/ldap/current/SUNWisp

% cp /etc/opt/SUNWisp/ldap/sunds/default/dsserv.acl.conf \ /etc/opt/SUNWconn/ldap/current/dsserv.acl.con

% cp /etc/opt/SUNWisp/ldap/sunds/default/mapping/radius.mapping \ /etc/opt/SUNWconn/ldap/current/mapping
```

5. **Start the directory services server by entering:**
   ```
   # /etc/init.d/dsserv start
   ```

Avoiding Schema Dependence

Because the Solaris ISP Server schema is still evolving, developing code that depends on it could make your program hard to maintain. Future releases of the product may have schema changes that are not compatible with the current schema. Any program that uses schema-dependent code, such as hard-coded object class or attribute names or assuming certain directory information tree structure, would have to be modified and recompiled with the new schema specifics.

To avoid this problem as much as possible, isolate any schema specifics in your code. When you need a specific detail, such as the distinguished name of a configuration entry, use the ISP Directory Information API to get it. If you need information that IDIA does not provide, use the LDAP client library and isolate the functionality in separate functions or classes in your code and reuse them.

ISP Directory Information API

The ISP Directory Information API provides C and Java programming language access to the directory services. The functions return information specific to the directory information tree (DIT) used by Solaris ISP Server. By default, the header file is located at `/opt/SUNWisp/include/isp_dir_api.h`, and the library is located at `/opt/SUNWisp/lib/libispdir.so.1`. The Java package is `com.sun.isp.idia`.

The directory information API includes the following:

`ispGetLdapInfo`(3X)	This C functions provides the distinguished name and password for binding to the LDAP server with access to a particular region of the directory information tree (DIT).
`ispGetLdapServers`(3X)	This C function provides the names and port numbers of LDAP directory servers configured on the network.
`ispGetTopDn`(3X)	This C function provides the distinguished name of the root domain (top-level domain entry in the DIT, under which Solaris ISP Server information is stored).
`ispLdapService`(3X)	This Java class provides information on LDAP servers configured on the network. Various class methods return the root domain entry in the DIT

and distinguished names and passwords for binding to the directory.

LDAP Client Library

The LDAP client library is an implementation of the LDAP v3 standard. It provides support for client applications communicating with an LDAP server such as Sun Directory Services.

By default, the header files are located at `/usr/include/ldap.h` and `/usr/include/lber.h`, and the library is located at `/usr/lib/libldap.so.3`. The LDAP client library man pages are located in `/usr/share/man`, in section 3.

Solaris ISP Server Object Classes

This section contains a list of the object classes added to the base schema to support Solaris ISP Server. Attributes listed as mandatory must have a value entered when the entry is created. Object classes are listed in alphabetical order.

`ispAdministrator` Class

Purpose: Defines an entry representing an administrator of ISP services and networks. The `ispAdministrator`'s relative distinguished name is the `commonName` attribute and its value. Its superior object is `ispSubscriber`.

TABLE 6–1 `ispAdministrator` Attributes

Attribute name	Mandatory?	Schema	Purpose
`associatedDomain`	No	Base	The domain with which this administrator is associated. Reserved for future Solaris ISP Server functionality.
`commonName`	Yes	Base	The name of the administrator described by the entry, in the form Firstname Lastname (userid).
`description`	No	Base	An arbitrary description of the administrator.

TABLE 6–1 `ispAdministrator` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`gidNumber`	No	Base	A UNIX group-ID. Specifies the gid of files created via FTP by this user.
`homeDirectory`	No	Base	The file system location of the home directory of the administrator described by the entry. (Not used by Solaris ISP Server.)
`ispAdministeredService`	No	Extension	The distinguished names of services this administrator is authorized to manage.
`ispContentDirectory`	No	Extension	A directory location where content belonging to the administrator is located. For virtually-hosted FTP or Web services, the path to user content relative to the `ispRootDirectory`.
`labeledURI`	No	Base	The Uniform Resource Identifier and label associated with the Web page of this administrator.
`mail`	No	Base	The advertised electronic mail address of the administrator.
`objectClass`	Yes	Base	The object class of the entry (`ispAdministrator`).
`ou`	No	Base	The organizational unit to which the entry belongs. In this release of Solaris ISP Server, the *Administrator* node under the root domain.
`surname`	Yes	Base	The family name of the administrator
`userCertificate`	No	Base	A certificate containing the public key of the administrator.
`userid`	Yes	Base	The login name of the administrator.

TABLE 6–1 `ispAdministrator` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`userPassword`	Yes	Base	The password of the administrator.
`uidNumber`	No	Base	A UNIX user-ID. Specifies the uid of files created via FTP by this user.

`ispManagedService` Class

Purpose: Defines an entry representing an ISP service that is managed by Sun Internet Administrator. This object class is reserved for use by Sun Internet Administrator. The `ispManagedService`'s relative distinguished name is the `host` attribute and its value.

TABLE 6–2 `ispManagedService` Attributes

Attribute name	Mandatory?	Schema	Purpose
`associatedName`	No	Base	The distinguished name of the top-level service entry for this service.
`commonName`	No	Base	The user-friendly name of a service, for display in the GUI of Sun Internet Administrator
`host`	Yes	Base	The fully-qualified name of the host where this service is installed.
`ispCategory`	No	Base	The type of user interface supported by this service. See the `ispCategory` attribute section for details.
`ispImageFile`	No	Extension	The name of a GIF image file containing the icon for this service.
`ispServiceLocation`	No	Extension	The path to the X-based administration application for this service.

TABLE 6–2 `ispManagedService` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`ispParameterizedOperation`	No	Extension	Information on a supported command-line utility that accepts parameters.
`ispServlets`	No	Extension	The fully-qualified Java class name of a servlet used in this service's administration user interface.
`ispServletClasspath`	No	Extension	The Java classpath for classes required by this service's administration user interface.
`ispSupportedOperation`	No	Extension	Information on a supported command-line utility that accepts no parameters.
`ispVersion`	No	Extension	The release number (major.minor) of the service described by the entry.
`labeledURI`	No	Base	The path to the main GUI page of a service. For a 3-tier service, enter a path relative to the document root. For a 2-tier service, enter the complete URL.
`objectClass`	Yes	Base	The object class of the entry (`ispManagedService`).

`ispService` Class

Purpose: Defines an entry representing a Solaris ISP Server software component. The `ispService`'s relative distinguished name is the `ispVersion` attribute and its value.

TABLE 6–3 `ispService` Attributes

Attribute name	Mandatory?	Schema	Purpose
`associatedDomain`	No	Base	The domain with which this service is associated.
`commonName`	Yes	Base	The name of the service (not used in Solaris ISP Server).
`description`	No	Base	An arbitrary description of the service.
`host`	No	Base	The fully-qualified name of the host where the service is installed.
`ispDirectoryRoot`	No	Extension	A directory prefix to a location on the file system where a domain's content is virtually-hosted. Used by Sun Internet FTP Server and Sun WebServer when in a virtual host configuration.
`ispPrivateData`	No	Extension	Software component password information for use by Sun Internet Administrator. This attribute is protected by ACLs from access by a user other than the directory root and Sun Internet Administrator.
`ispServiceContext`	No	Extension	A CORBA naming context used by Sun Internet FTP Server and Sun Internet News Server.
`ispServiceLocation`	No	Extension	A CORBA stringified object reference to the service administration server. (Used by Sun Internet FTP Server and Sun Internet News Server.)

TABLE 6–3 `ispService` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`ispSupplementaryInformation`	No	Extension	Arbitrary information about the service. Reserved for Solaris ISP Server service-specific needs.
`ispVersion`	Yes	Extension	The release number (major.minor) of the service described by the entry.
`labeledURI`	No	Base	The path to the servlets for a three-tier GUI.
`mail`	No	Base	The advertised electronic mail address of the user. Not used by Solaris ISP Server.
`objectClass`	Yes	Base	The object class of the entry (`ispService`).
`userCertificate`	No	Base	A certificate containing the public key of the user.
`userPassword`	No	Base	The password of the entry, used for binding to the directory

`ispSubscriber` Class

Purpose: Defines an entry representing a subscriber (customer) of the ISP. The `ispSubscriber`'s relative distinguished name is the `commonName` attribute and its value. Its superior object is `inetOrgPerson`.

If you are using the RADIUS server that comes with Sun Directory Services, overlay the `ispSubscriber` objects with the `remoteUser` object class.

TABLE 6–4 `ispSubscriber` Attributes

Attribute name	Mandatory?	Schema	Purpose
`associatedDomain`	No	Base	The domain with which this subscriber is associated.
`commonName`	Yes	Base	The name of the subscriber described by the entry, in the form Firstname Lastname (userid).
`gidNumber`	No	Base	A UNIX group-ID. Specifies the gid of files created via FTP by this user.
`homeDirectory`	No	Base	The file system location of the home directory of the subscriber described by the entry. (Not used by Solaris ISP Server.)
`host`	No	Base	The fully-qualified name of the host. Not used by Solaris ISP Server.
`ispAuthorizedServices`	No	Extension	The distinguished names of services the subscriber is authorized to use.
`ispContentDirectory`	No	Extension	A directory location where content belonging to a subscriber is located. For virtually-hosted FTP and Web services, the path relative to the `ispRootDirectory`.
`labeledURI`	No	Base	The Uniform Resource Identifier and label associated with the Web page of this subscriber.
`mail`	No	Base	The advertised electronic mail address of the subscriber.
`objectClass`	Yes	Base	The object class of the entry (`ispSubscriber`).
`ou`	No	Base	The organizational unit to which the subscriber belongs (in Solaris ISP Server, the *People* node under a domain entry).
`surname`	Yes	Base	The family name of the subscriber.
`uidNumber`	No	Base	A UNIX user-ID. Specifies the uid of files created via FTP by this user.
`userCertificate`	No	Base	A certificate containing the public key of the subscriber.

TABLE 6–4 `ispSubscriber` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`userid`	Yes	Base	The login name of the subscriber.
`userPassword`	Yes	Base	The password of the subscriber.

Solaris ISP Server Attributes

This section describes the attributes used by object classes added to the basic schema for Solaris ISP Server. Some of these attributes are a part of the base schema, but are included here for ease of use. Attributes created for the Solaris ISP Server schema extension begin with "isp."

Attributes defined in the schema have one of the following syntaxes:

- `bin`: binary
- `ces`: case-exact string

  A case-sensitive alphanumeric string.
- `cis`: case-ignore string

  An alphanumeric string, not case-sensitive.
- `dn`: distinguished name
- `protected`: encrypted

  A value that has been encrypted using `crypt`
- `int` or `long`: integer
- `tel`: telephone number
- `utctime`: UTC time

The following list presents attributes in alphabetical order.

`associatedDomain` Attribute

Summary: `cis`, multi-valued, base schema

Purpose: Specifies the domain with which the object described by the entry is associated. For the OSI tree entry of a domain, must contain the name (in dot notation, for example, eng.sun.com) of the corresponding DC tree entry.

`associatedName` Attribute

Summary: `dn`, base schema

Purpose: Specifies the distinguished name of an entry associated with this entry. For the DC tree entry of a domain, must contain the distinguished name of the corresponding OSI tree entry.

`commonName` Attribute

Alternate name: `cn`

Summary: `cis`, multi-valued, base schema

Purpose: Specifies the full name of the object described by the entry.

Subscriber and administrator entries use the form *Firstname Lastname (uid)*. For example, user John Smith who uses the login name jsmith would have an entry in the directory. Its `commonName` attribute would contain the value `John Smith (jsmith).`

The `ispManagedService` object class uses this attribute for the user-friendly name displayed in the Sun Internet Administrator GUI.

`description` Attribute

Summary: `cis`, multi-valued, base schema

Purpose: Specifies an arbitrary description of the entry object.

`gidNumber` Attribute

Summary: `long`, single-valued, base schema

Purpose: Specifies the UNIX group-ID of files created via FTP by this user.

`homeDirectory` Attribute

Summary: `ces`, single-valued, base schema

Purpose: Specifies the file system location of the home directory of the user described by the entry. This attribute is not used by Solaris ISP Server, but is left for general information required by the customer.

`host` Attribute

Summary: `cis`, multi-valued, base schema

Purpose: Specifies the name of the machine associated with or managed by the object described by the entry. When used with Solaris ISP Server object entries, the host attribute must contain the fully-qualified host name.

`ispAdministeredService` Attribute

Summary: `dn`, multi-valued, Solaris ISP Server schema extension

Purpose: Specifies the services (by distinguished names of the top-level service entries) that an administrator can manage. The top-level entry for a service is an `ispService` object under the root domain entry. For example, under ou=services,o=sun,c=us where sun.com is the root domain entry.

`ispAuthorizedServices` Attribute

Summary: `dn`, multivalued, Solaris ISP Server schema extension

Purpose: Specifies the services (by distinguished name) that an `ispSubscriber` is authorized to use.

`ispCategory` Attribute

Summary: `cis`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies the category of graphical user interface supported by the service described by the entry. Acceptable values are:

- `2tier`, indicating that the user interface is Web-based and uses the two-tier architecture supported by Sun Internet Administrator.
- `3tier`, indicating that the user interface is Web-based and uses the three-tier architecture supported by Sun Internet Administrator.
- `CLI`, indicating that the user interface is a command-line utility.
- `X`, indicating that the user interface is an X-based program.

`ispContentDirectory` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies a directory location (which may be distinct from that in the `homeDirectory` attribute) where content belonging to a user is located. Used by `ispAdministrator` and `ispSubscriber` classes. For a user with virtually-hosted FTP services, must contain the path (relative to the `ispDirectoryRoot` of the domain) to the user’s FTP content.

`ispDirectoryRoot` Attribute

Summary: ces, single-valued, Solaris ISP Server schema extension

Purpose: Specifies a root directory for an ISP service, usually on a per-domain basis.

`ispImageFile` Attribute

Summary: `cis`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies the name of a file containing the image described by the entry. The `ispManagedService` object class uses this attribute to specify its GIF-format icon file.

`ispPrivateData` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies software component password information used by Sun Internet Administrator. This attribute is protected by ACLs from access by users other than the Sun Directory Services root administrator.

`ispServiceContext` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies a service context for use by an ISP service. For Sun Internet FTP Server and Sun Internet News Server, this is a CORBA naming context.

`ispServiceLocation` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies the location of the `ispService` object described by the entry.

Sun Internet FTP Server and Sun Internet News Server use this attribute in an `ispService` object to store a CORBA stringified object reference to their administration servers. Sun Internet Administrator uses this attribute in an `ispManagedService` object to store the path to an X-based user interface application.

`ispServlets` Attribute

Summary: `ces`, multivalued, Solaris ISP Server schema extension

Purpose: Specifies the fully-qualified Java class name of a servlet used in an `ispManagedService`'s administration user interface. For each servlet used, assign an `ispServlets` attribute with the following value:

1. The path to the servlet, relative to the document root of the administration Web server (part of Sun Internet Administrator)
2. The fully-qualified Java class name of the servlet
3. Any required servlet arguments, listed by name and value

`ispServletClasspath` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies the Java classpath for classes required by an `ispManagedService`'s administration user interface. A Java classpath may contain several paths; separate them with colons (:).

`ispParameterizedOperation` Attribute

Summary: `ces`, multivalued, Solaris ISP Server schema extension

Purpose: Specifies information about a command-line function that takes parameters. For an `ispManagedService` object, three space-delimited fields of information are required:

1. The complete path to the command-line utility
2. The complete path to a help file for the utility, or `NONE` if there is no help information for the utility.
3. An arbitrary string, containing the space-delimited parameters required by the utility. If parameters are accepted from the user on the command line, this field holds the string `SOME`.

`ispSupplementaryInformation` Attribute

Summary: `cis`, multivalued, base schema

Purpose: Holds additional information concerning the object described by the entry. In object classes that extend the base schema, this attribute is reserved for Solaris ISP Server service-specific needs.

`ispSupportedOperation` Attribute

Summary: `ces`, multivalued, Solaris ISP Server schema extension

Purpose: Stores information on a supported command-line utility that accepts no parameters. For an `ispManagedService` object, three space-delimited fields of information are required:

1. The complete path to the command-line utility
2. The complete path to a help file for the utility, or `NONE` if there is no help information for the utility.
3. The string `NONE`.

`ispVersion` Attribute

Summary: `ces`, single-valued, Solaris ISP Server schema extension

Purpose: Specifies the release version of the `ispService` object described by this entry. Solaris ISP Server uses the form major.minor for the version attribute.

`labeledURI` Attribute

Alternate name: `labeledURL`

Summary: `ces`, multi-valued, base schema

Purpose: Specifies a Uniform Resource Identifier (URI) and label associated with the object described by the entry.

`mail` Attribute

Alternate name: `preferredRfc822Originator`

Summary: `cis`, multi-valued, base schema

Purpose: Specifies the advertised electronic mail address, in RFC822 format, of the object described by the entry. Sun Internet FTP Server uses this attribute as an error-reporting address.

`objectClass` Attribute

Summary: `cis`, multivalued, base schema

Purpose: Specifies the object class of the type of entry.

`ou` Attribute

Alternate name: `organizationUnitName`

Summary: `cis`, base schema

Purpose: Specifies the name of the organization unit to which the object described by the entry belongs.

`surname` Attribute

Alternate name: `sn`

Summary: `cis`, base schema

Purpose: Specifies the surname of the person described by the entry.

`uidNumber` Attribute

Summary: `long`, single-valued, base schema

Purpose: Specifies the UNIX user-ID of files created via FTP by this user.

`userCertificate` Attribute

Summary: `bin`, base schema

Purpose: Specifies a certificate containing the public key of the user described by the entry.

`userid` Attribute

Alternate name: `uid`

Summary: `cis`, multi-valued, base schema

Purpose: Specifies the login name of the user described by the entry.

`userPassword` Attribute

Summary: `protected`, multi-valued, base schema

Purpose: Specifies the password for the entity described by the entry.

Schema Changes for Sun™ Internet Mail Server

If you configure your Solaris ISP Server installation to interoperate with Sun Internet Mail Server, a few additions are made to your schema. The schemas of both the main directory and the slave installation that supports mail must be identical. This new object class and its attributes support calendaring, but change no functionalty on the Solaris ISP Server side.

`IMCalendarUser` Object Class

Purpose: Defines an entry that stores calendaring information.

TABLE 6–5 `IMCalendarUser` Attributes

Attribute name	Mandatory?	Schema	Purpose
`IMCalendarHost`	Yes	SIMS	The fully-qualified host name (the machine where the calendar server is running).
`IMCalendarName`	No	SIMS	The name of the calendar.
`objectClass`	Yes	Base	The object class of the entry (`IMCalendarUser`, but there may be others)

TABLE 6–5 `IMCalendarUser` Attributes *(continued)*

Attribute name	Mandatory?	Schema	Purpose
`userid`	Yes	Base	The login name of the calendar user.
`userPassword`	Yes	Base	The calendar user's password (shared with SIMS/IMAP login).

`IMCalendarHost` Attribute

Summary: `cis`, single-valued, SIMS schema

Purpose: The fully qualified calendar server host name.

`IMCalendarName` Attribute

Summary: `ces`, single-valued, SIMS schema

Purpose: The name of the user's calendar.

CHAPTER **7**

Integrating Existing Service Applications

Solaris ISP Server™ is designed to allow integration of your existing services with Sun™ Internet Administrator™. When you integrate these services, they gain the benefits of centralized remote administration and administrator account management provided by that product. Because Sun Internet Administrator manages administrator access, integrating an application with it is an easy way to add the security of administrator authorization to an existing service application.

Topics in this chapter include:

General Steps to Integrating an Existing Service

Sun Internet Administrator integrates and manages Web-based, X-based, and command-line administration interfaces using the following, general steps.

1. Install the service application on a computer that is accessible by Sun Internet Administrator via the network. This host computer must have the Solaris ISP Server platform component (SUNWisp) installed on it, to provide platform extensions that Sun Internet Administrator uses.
2. Run the `mcreg(1M)` command on the service host computer with the correct parameters for this type of service. The form of the `mcreg` command is different

for each type of service and is fully described later in this chapter as well as on the man page for the command. You must have root access to run `mcreg`.

Note - You must run `mcreg` again if you change the URL or other service configuration information (for example, if you change the port where an administrative Web server is running). If you run `mcreg` more than once for a service (on a single host machine), the later information overwrites the information recorded previously.

---

3. In Sun Internet Administrator, register the service host and select the new service.
4. Also in Sun Internet Administrator, create administrators with access to the new service or modify existing administrators to grant them access.

   You must be an administrator with rights to manage Sun Internet Administrator to perform these tasks.

Integrating X-Based Services

When the service application is installed on a computer in the network, run the `mcreg(1m)` command to store information about the application. This information is used by Sun Internet Administrator to correctly handle the presentation and launch of the administration GUI. The information you need for the `mcreg` command is:

- The component identifier, a unique string that identifies the service application. Because they are unique, Solaris ISP Server recommends using package names as component identifiers. The service application developer chooses the component identifier.
- A user-friendly name, appropriate for presentation to the administrator. This is the service name that Sun Internet Administrator displays in its *Manage Services* screen, where the administrator accesses all services. The service application developer chooses the application name.
- The version number of the application being registered. The recommended format for the version number is *major.minor* (for example, 1.2). The service application developer chooses the version number.
- The complete file system path to the X-based administration interface executable on the service host.
- The user name under which the administration interface runs. This identifies the UNIX UID that Sun Internet Administrator uses to invoke the program.
- The group name under which the administration interface runs. This identifies the UNIX GID that Sun Internet Administrator uses to invoke the program.

When you have all required information, obtain root access and register the information as shown below. Run this command on the computer where the service administration interface is installed.

# mcreg –c *componentID* –n *name* –v *version* –x *X_path* –u *user_name* –g *group_name*

See the `mcreg(1m)` man page for an example of this form of the command.

After running `mcreg`, you can log into Sun Internet Administrator as a console administrator and register the new application for management. If you want other administrators to have rights to manage the new application, edit their administrator access to include it. The Sun Internet Administrator on-line help has full instructions on performing these tasks.

Integrating Command-Line Programs

To integrate a command-line administration program with Sun Internet Administrator, run the `mcreg(1m)` command to store information about the application. This information is used by Sun Internet Administrator to correctly handle the presentation and launch of the command. The information you need for the `mcreg` command is:

- The component identifier, a unique string that identifies the service application. The service application developer chooses the component identifier.
- A user-friendly name, appropriate for presentation to the administrator. This is the service name that Sun Internet Administrator displays in its *Manage Services* screen, where the administrator accesses all services. The service application developer chooses the application name.
- The version number of the application being registered. The recommended format for the version number is *major.minor* (for example, 1.2). The service application developer chooses the version number.
- The complete path to the command-line program, plus information about its parameters and documentation. Enclose this command option in quotes ("). There are three fields:
  - The full path to the executable, with any static parameters. For example, `-p ''/usr/bin/ps -ef''`. This field is required.
  - A **`-a`** indicates that this command takes parameters from the user at run time. This field is optional
  - A **`-h helpfile`** indicates the on-line documentation associated with this command. This field is optional.
- The user name under which the administration interface runs. This identifies the UNIX UID that Sun Internet Administrator uses to invoke the program.

- The group name under which the administration interface runs. This identifies the UNIX GID that Sun Internet Administrator uses to invoke the program.

When you have all required information, obtain root access and register the information as shown below. Run this command on the computer where the service administration interface is installed.

# mcreg `–c` *componentID* `–n` *name* `–p` "*prog_path* [`–a`] [`–h` *help_file*]"... `–v` *version* `–u` *user_name* `–g` *group_name*

See the `mcreg(1m)` man page for examples of this and other forms of the command.

To record information about multiple command-line operations, enter multiple `–p` arguments.

After running `mcreg`, you can log into Sun Internet Administrator as a console administrator and register the new application for management. If you want other administrators to have rights to manage the new application, edit their administrator access lists to include it. The Sun Internet Administrator on-line help has full instructions on performing these tasks.

Integrating Two-Tier Web-Based Applications

A two-tier Web-based application is an existing application whose administration interface is accessed through a Web browser. It can be implemented in HTML, CGI, or Java Applets and Servlets (or some combination). Such an interface requires a Web server installed and running on the computer where the service is installed.

Because this interface is accessible via URLs, there are some security issues to consider. The interface should be protected by the Web server's ACLs so only authorized administrators can access it. To take advantage of the administrator management provided by Sun Internet Administrator, you must use Sun™ WebServer™ and configure it with the same ACLs used by the Sun Internet Administrator administration Web server (see "Configuring for Administrator Account Coordination" on page 89).

To secure the connection between the browser and the administration interface, consider secure HTTP (HTTPS) or SKIP. See the Sun WebServer on-line help and the SunScreen SKIP man pages (`/opt/SUNWicp/man`) for instructions on how to configure these security tools.

Registering Information for a Two-Tier Web-Based Application

When the service application is installed on a computer in the network, run the `mcreg(1m)` command to store information about the application. This information is used by Sun Internet Administrator to correctly handle the presentation and launch of the administration GUI. The information you need for the `mcreg` command is:

- The component identifier (*componentID*), a unique string that identifies the service application. Because they are unique, Solaris ISP Server recommends using package names as component identifiers. The service application developer chooses the component identifier.
- A user-friendly *name*, appropriate for presentation to the administrator. This is the service name that Sun Internet Administrator displays in its *Manage Services* screen, where the administrator accesses all services. The service application developer chooses the application name.
- The version number of the application being registered. The recommended format for the version number is *major.minor* for example, 1.2). The service application developer chooses the version number.
- The *URL* to the entry point of the Web-based administration interface. Enter the absolute (not a relative) path. The URL should use the fully qualified domain name of the service host.

When you have all required information, obtain root access and register the information as shown below. Run this command on the computer where the service administration interface is installed.

# mcreg `–c` *componentID* `–n` *name* `–v` *version* `–w` *URL*

See the `mcreg(1m)` man page for an example of this form of the command.

After running `mcreg`, you can log into Sun Internet Administrator as a console administrator and register the new application for management. If you want other administrators to have rights to manage the new application, edit their administrator access lists to include it. The Sun Internet Administrator on-line help has full instructions on performing these tasks.

Configuring for Administrator Account Coordination

If the two-tier service application uses Sun WebServer for its administration interface, it is possible to configure the server's ACLs to use the same administrator login information as Sun Internet Administrator. Thus, you have a single set of administrator accounts (in the directory services) to manage. This simplifies management of these accounts and enhances your ability to respond to security risk situations.

You must first install Sun WebServer on the service host (the computer where the service application is installed), and arrange the interface documents and files within the document tree of the server. Then, configure SWS as follows:

1. Create a realm in the default Web site of the SWS instance:

   ```
   # htrealm add  –i instance –h hostname  –r realmname  –s ISPADMIN –d
           ComponentID-VersionNo
   ```

   Where

   - *instance* is the name of the httpd instance being configured.
   - *hostname* is the name of the host containing the realm.
   - *realmname* is the name of the realm you are creating.
   - *ISPAdmin* is the source of the realm. Enter ISPADMIN, indicating that these are Sun Internet Administrator administrators.
   - *ComponentID-VersionNo* is the directory where the administration GUI files (HTML and others) are located.

2. Add ACL protection to the URL where the administration GUI for this service resides:

   ```
   # htaccess add  –i instance  –h hostname –U URI –r realname –s BASIC
   ```

   Where

   - *instance* is the name of the httpd instance being configured.
   - *hostname* is the name of the computer where the Web server is running.
   - *realmname* is the name of the realm you are creating.
   - *BASIC* is the authentication scheme wanted. Enter BASIC.

3. Restart the default site, if it is already running. Use the Sun WebServer administration GUI to perform this task. Full instructions are available in the on-line help.

APPENDIX **A**

Configuring a Virtual Host

This appendix explains the steps you need to take to configure a virtual domain or virtual host in the Solaris ISP Server™ directory service. Only the configuration steps for the directory service are covered; specific configurations for adding IP addresses or domain names and for ISP services such as FTP are not covered here.

The procedures in this appendix will assist you in creating a virtual domain that has its own set of subscribers who use a Sun Internet FTP Server virtual FTP site and a Sun™ WebServer™ virtual web site.

Topics include

Before You Begin

Before you configure the directory service to support a virtual host, you should plan for or implement the parts of the virtual host not related to the directory.

The first step in the configuration is registering a domain name and assigning an IP address. If you plan to use a Sun Internet FTP Servervirtual FTP site, you must assign a unique IP address to the virtual host to be used for FTP. If you do not require a virtual FTP site with domain-specific users, you may be able to use name based virtual hosting (where several domain names share the same IP address).

You must decide whether the virtual domain has subscribers and how they will use the system. The way that subscribers can log into the virtual host (or not) affects how you refer authentication requests in the directory and how you configure the location of virtual host data on disk.

Often a virtual host does not have unique subscribers (subscribers are authenticated in the domain of the hosting ISP or the subscribers do not actually store content at the ISP); in this case you may decide to ignore subscriber information or refer to a subscriber base (an `ou=People` node) that is not specific to the virtual host. In this case, you only need to plan the location of the virtual host data on disk and configure services accordingly.

Sometimes a virtual host requires only a single point of contact. An administrator from the customer's organization is the only person who actually logs in to store or remove data from a web site or FTP site. In this case, you may want to create a subscriber base unique to the virtual host, but in which the user's "home directory" is the root of the virtual web or FTP site rather than a personal directory.

Some virtual domains contain subscribers who are unique to that domain, who are not ISP subscribers, and who have personal web or FTP areas (for example, a small business that has its intranet hosted at your ISP). In this case, you will need to create a root directory for the FTP and web sites and a location for user personal directories. The virtual domain in the OSI tree will require an `ou=People` node with its own subscribers, and the hosts in the DC tree will point to this node to authenticate users. Each user entry will have an `ispContentDirectory` attribute naming the location of the personal directory for FTP and web site access.

Finally, if any subscriber will log in to the domain to store Sun Internet FTP Server data or Sun WebServer site content, you will need to create an `ispService` entry in the OSI tree that names the location of the content. The `ispService` object class `ispDirectoryRoot` attribute determines what part of the file system will be available to users when they authenticate in the virtual domain. If there are many subscribers with personal directories, you will want to designate a personal directory in each user's `ispContentDirectory` attribute.

Creating Directory Entries

The following procedures document how to create the entries in the directory to support a virtual host. The procedures show you how to create ldif files to use with `ldapmodify` to make the entries, but you may use the Deja tool instead. At the end of the section, there is an example ldif file with all of the entries you would make if you follow each procedure.

The examples in the procedures use the most complicated case: a virtual host with its own, unique set of subscribers who have personal directories for web and FTP content. The configuration you use may be less complicated, and may not require all of these procedures.

You only need to create these directory entries if you need to authenticate subscribers against the directory. If subscribers never authenticate with the FTP or web server (or any other service you may add that authenticates users against the directory), you do not need special entries in the directory.

The examples in the procedure will assume the following sample data:

- There is a virtual domain called `smallorg.com`.
- There is one virtual host in the domain, `www.smallorg.com` (111.112.113.14) , which subscribers use for web site content.
- The site is also configured as a Sun Internet FTP Server virtual FTP site so that users may upload content using FTP.
- The virtual domain is for an organization with multiple subscribers.
- Subscribers are authenticated only in the virtual domain, not as subscribers to the main ISP organization.
- The root directory of the virtual FTP site and web site is `/var/hosts/smallorg/public/`.
- Each subscriber has a directory directly beneath the root directory (`/var/hosts/smallorg/public/username/`).
- The name of the hosting ISP is “myISP,” and it is represented in the OSI tree in the directory as the naming context `o=myISP,c=US`.

Entries in the DC Tree

The entries in the DC tree allow services to map the virtual host DNS name to an area in the OSI tree where service configuration and subscriber data is stored.

By default, the DC tree contains a top level `dc=com` node representing the `.com` top-level DNS domain. Each host is represented by a chain of dc nodes that are children of the top node. For example, `www.sun.com` would be represented as `dc=www,dc=sun,dc=com`.

If you need to add hosts that are not in the `.com` domain, you will need to create another naming context (for example, a top-level `dc=net` node). See “Adding More Top-level Domains” on page 100 for more help with creating a new naming context.

▼ To Create the Domain in the DC Tree

1. **Determine the name of each virtual host subscribers may authenticate on.**

   You only need to configure the hosts that will use the directory to authenticate users.

In this example, we will use one host: `www.smallorg.com`

2. **If there is no naming context for the top-level domain for the virtual host, create one.**

   You can view the current naming contexts using Deja or by using the web gateway (if it is enabled) .

   If there is no `dc=com` domain, for example, you will need to create one before `www.smallorg.com` can be added. See “Adding More Top-level Domains” on page 100 for more help with creating a new naming context.

3. **Determine the distinguished name (dn) of the node in the OSI that contains the subscribers who use this host. This will be the `associatedName` attribute on the dc entry for the host.**

   This example assumes that the domain `smallorg.com` has its own unique set of subscribers. The entries may not exist in the directory yet, but they will be created later. The dn in the OSI tree will be `ou=SmallOrg,o=myISP,c=US`.

4. **Create a text file for the ldif entries.**

   For example, `/tmp/smallorg.ldif`.

5. **For each dc node you need to add, create an ldif entry naming the dn (dc=name), the `objectclass` (`domain`), `dc`, and the `associatedName` in the OSI tree.**

   For our example, we need to add dc nodes for smallorg (beneath com) and www (beneath smallorg). The `dc=www` node needs to point to the proper place in the OSI tree so that the correct users can log in to that host.

```
dn: dc=smallorg,dc=com
   dc: smallorg
   objectclass: domain

   dn: dc=www,dc=smallorg,dc=com
   dc: www
   objectclass: domain
   associatedname: ou=SmallOrg,o=myISP,c=US
```

6. **Save the file.**

7. **You may run `ldapadd` now to add the entries, or continue with the other procedures and add all of the required ldif entries to the file first.**

   To run `ldapadd` on the host where the Solaris ISP Server directory server is running:

```
% cd /opt/SUNWconn/bin
% ./ldapadd -D cn=admin,o=myISP,c=US -w secret -f /tmp/smallorg.ldif
```

8. **Continue with the procedure in "Entries in the OSI Tree" on page 95.**

Entries in the OSI Tree

The OSI tree must contain entries for the subscribers who will be authenticated on the virtual host.

The virtual host or virtual domain does not need to have its own set of subscribers; the virtual host services can authenticate users against the general ISP subscriber base. If you wish to have users authenticated against the general ISP subscriber base, make the `associatedName` attribute on the `dc` entries in the DC tree point to the top DN in the OSI tree (for example, `ou=myIsp,c=US`).

If you create a new domain in the OSI tree, you need to create an `organizationalUnit` (`ou`) node naming the domain with an `associatedDomain` attribute that points to the proper domain or host name in the DC tree. Below the virtual domain `ou`, you need to create `ou` nodes for People (to contain `ispSubscriber` entries), Groups, and Services (to contain `ispService` entries). Populating the People and Services portions of the tree is explained in "Entries for Users for the Domain" on page 97 and "Entries for Services in the Domain" on page 98 below.

The procedure below uses a virtual host with its own set of subscribers. Since these subscribers are unique to the virtual domain, we need to create an `ou` node in the OSI tree for the domain and set its `associatedDomain` attribute to the name of the virtual domain in the DC tree.

▼ To Create the Domain in the OSI Tree

1. **Determine the distinguished name for the domain in the OSI tree. The distinguished name includes the `ou` node for the domain, and all of the parents of that node in the tree.**

   In this procedure we will create a domain directly below a top distinguished name of `o=myISP,c=US`. The dn of the domain is `ou=SmallOrg,o=MyISP,c=US`.

   Since we have only one host in the example, we could be more specific and use `ou=www,ou=SmallOrg,o=MyISP,c=US`. By only specifying the domain, it is possible to add more virtual hosts to the domain later that use the same subscribers (for example, if `mail.smallorg.com` is created later).

2. **Determine the dn of the virtual host in the DC tree. This will be used in the `associatedDomain` attribute.**

   For example, `dc=smallorg,dc=com`.

3. **Create a text file for the ldif entries.**

   For example, `/tmp/smallorg.ldif`.

4. **Create an ldif entry for the domain specifying the `dn`, its object class attributes, the `ou`, and the `associatedDomain`.**

   The object classes are always `organizationalUnit` and `domainRelatedObject`.

   For this example, `dn=ou=SmallOrg,o=MyISP,c=US` and the `associatedDomain` is `smallorg.com` (note that the `associatedDomain` is specified as a domain name and not as a distinguished name).

   Here is what would be entered into the ldif file:

```
dn: ou=SmallOrg,o=MyISP,c=US
   ou: SmallOrg
   associatedDomain: smallorg.com
   objectClass: organizationalUnit
   objectClass: domainRelatedObject
   objectClass: top
```

5. **Create ldif entries for the `People`, `Groups`, and `Services` nodes in the domain.**

   These nodes all have `organizationalUnit` object classes, and are used to contain other data for the virtual domain.

   The following would be added to the end of the example ldif file:

```
dn: ou=People,ou=SmallOrg,o=MyISP,c=US
   ou: People
   objectClass: organizationalUnit

   dn: ou=Groups,ou=SmallOrg,o=MyISP,c=US
   ou: Groups
   objectClass: organizationalUnit

   dn: ou=Services,ou=SmallOrg,o=MyISP,c=US
   ou: Services
   objectClass: organizationalUnit
```

6. **Save the file.**

7. **You may run** `ldapadd` **now to add the entries, or continue with the other procedures and add all of the required ldif entries to the file first.**

   To run `ldapadd` on the host where the Solaris ISP Server directory server is running:

```
% cd /opt/SUNWconn/bin
% ./ldapadd -D cn=admin,o=myISP,c=US -w secret -f /tmp/smallorg.ldif
```

Entries for Users for the Domain

After you create the domain in the OSI tree, you can add entries for the users of that domain below the `ou=People` node. For detailed information on creating users, see Chapter 5.

The `ispSubscriber` attributes specific to the virtual host are `ispContentDirectory` and `ispAuthorizedServices`.

`ispContentDirectory` specifies the user's directory on the virtual host. The directory is relative to the value of the `ispDirectoryRoot` of the service the user accesses (see "Entries for Services in the Domain" on page 98 for information on this attribute). The `ispContentDirectory` is used by the Sun Internet FTP Server and the Sun WebServer to determine where to store content that a user puts on the virtual host.

If a virtual host only has one authorized user (an administrative account for a hosted web site, for example), you may wish to create a single subscriber in the domain who only has access to the web site directory. In this case, the `ispContentDirectory` could be "`/`" to indicate that when the user logs in to the Sun Internet FTP Server his or her root directory will be the `ispDirectoryRoot` of the FTP site.

`ispAuthorizedServices` specifies the distinguished names of services in the virtual domain that the user can access (if authentication is required). An example dn value for this attribute is `ispVersion=2.1,ou=SUNWhttp,ou=Services,ou=SmallOrg,o=MyISP,c=US`.

The following example shows an ldif entry for a subscriber that could be part of the `www.smallorg.com` domain described in other sections:

```
dn: cn=John Doe (jdoe),ou=People,ou=SmallOrg,o=MyISP,c=US
objectClass: top
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: ispSubscriber
cn: John Doe
sn: Doe
```

```
givenName: John
userId: jdoe
userPassword: secret
uidNumber: 19995
gidNumber: 101
ispContentDirectory: jdoe/public_html
ispauthorizedServices: ispVersion=1.0,ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
mail: jdoe@myisp.com
```

Entries for Services in the Domain

Each service that the users in the virtual domain can log in to requires an `ispService` entry beneath the `ou=Services` node in the domain. The only service bundled with the Solaris ISP Server software that can authenticate users in virtual domains is Sun Internet FTP Server, so that is the only service covered in this section.

The Sun WebServer uses the `ispDirectoryRoot` of a Sun Internet FTP Server entry to determine where to store or find user personal content. If users have personal web site directories, create an `ispService` entry for a Sun Internet FTP Server service even if a virtual FTP site has not been created.

▼ To Create a Sun Internet FTP Server Service Entry

1. **Determine the directory root used by the Sun Internet FTP Server or Sun WebServer service on the virtual host.**

   This example will use `/var/hosts/smallorg/public/`. Beneath this directory there may be a `./pub` directory for anonymous FTP, web site content, and user directories in `./`*username*`/public_html`.

2. **Create a text file for the ldif entries.**

   For example, `/tmp/smallorg.ldif`.

3. **Add an `ou` entry for SUNWftp to the `Services` node for the domain.**

```
dn: ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
   ou: SUNWftp
   objectClass: organizationalUnit
```

4. **Add an `ispService` entry beneath the `SUNWftp` entry. The dn of this entry is the `ispVersion` attribute plus the dn of the `SUNWftp` entry you just added.**

   The `ispVersion` attribute is always “1.0.”

```
dn: ispVersion=1.0,ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
   ispVersion: 1.0
   cn: SUNWftp
   objectClass: ispService
   ispDirectoryRoot: /var/hosts/smallorg/public
```

5. **Save the file.**

6. **Run** `ldapadd` **now to add the entries.**

   To run `ldapadd` on the host where the Solaris ISP Server directory server is running:

```
% cd /opt/SUNWconn/bin
% ./ldapadd -D cn=admin,o=myISP,c=US -w secret -f /tmp/smallorg.ldif
```

Example ldif File

For reference, the listing below shows the complete contents of an ldif file with all of the example entries used in this Appendix:

CODE EXAMPLE A–1 Complete ldif File to Add a Virtual Host

```
dn: dc=smallorg,dc=com
dc: smallorg
objectClass: domain

dn: dc=www,dc=smallorg,dc=com
dc: www
objectClass: domain
associatedName: ou=SmallOrg,o=myISP,c=US

dn: ou=SmallOrg,o=MyISP,c=US
ou: SmallOrg
associatedDomain: smallorg.com
objectClass: organizationalUnit
objectClass: domainRelatedObject
objectClass: top

dn: ou=People,ou=SmallOrg,o=MyISP,c=US
ou: People
objectClass: organizationalUnit

dn: ou=Groups,ou=SmallOrg,o=MyISP,c=US
ou: Groups
objectClass: organizationalUnit

dn: ou=Services,ou=SmallOrg,o=MyISP,c=US
ou: Services
objectClass: organizationalUnit

dn: cn=John Doe (jdoe),ou=People,ou=SmallOrg,o=MyISP,c=US
objectClass: top
objectClass: person
objectClass: inetOrgPerson
```

```
objectClass: organizationalPerson
objectClass: ispSubscriber
cn: John Doe
sn: Doe
givenName: John
userId: jdoe
userPassword: secret
uidNumber: 19995
gidNumber: 101
ispContentDirectory: jdoe/public_html
ispauthorizedServices: ispVersion=1.0,ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
mail: jdoe@myisp.com

dn: ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
ou: SUNWftp
objectClass: organizationalUnit

dn: ispVersion=1.0,ou=SUNWftp,ou=Services,ou=SmallOrg,o=MyISP,c=US
ispVersion: 1.0
cn: SUNWftp
objectClass: ispService
ispDirectoryRoot: /var/hosts/smallorg/public
```

Adding More Top-level Domains

The DC tree is configured with `dc=com` as the naming context, making `.com` the (only) top-level domain. You may need to add other naming contexts to support virtual hosts that are not in the `.com` domain.

The following procedure documents how to use the Sun Directory Services `dsadmintool` administration console to create an additional DC tree for the `.net` domain. The same procedure, of course, applies to any top-level domain.

The Sun Directory Services software supports up to four naming contexts in a data store, or three DC tree naming contexts and the OSI tree. If your ISP requires more than three top-level domain naming contexts, see the procedure "To Create an Additional Data Store" below.

▼ To Add a DC Tree for Another DomainTo Add a DC Tree for Another Domain

1. **Log in to the machine running the Solaris ISP Server directory service.**

2. **Run** `/opt/SUNWconn/sbin/dsadmintool`**.**

   `dsadmintool` is an X application. Make sure your DISPLAY environment variable is properly set if you are not at the console.

3. **Click *Data Store* in the *Sections* list to display the Data Store section.**

4. **Click the *Datastore* folder in the Data Store section to select it.**

5. **Choose *Modify Data Store* from the *Selected* menu.**

   The Modify Data Store dialog is displayed.

6. **Click the *More Suffixes* button.**

   A new *Additional Suffix* text field displays at the end of the list.

7. **Type "dc=" and the name of the top-level domain you want into the *Additional Suffix* field.**

   For example, to add a DC tree for the `.net` domain, enter **`dc=net`**.

8. **Click *OK* to apply the change and close the Modify Data Store dialog.**

9. **In the Sun Directory Services administration console window, click *Apply* to save the changes to the Data Store and restart the server.**

   A dialog will prompt you to ask if you want to restart the server. The new naming context will not be available until the server has been restarted. The remaining steps assume that you have restarted the server so that an entry can be added to the new naming context.

10. **Create an ldif file with a domain entry for the root of the new naming context.**

    For example:

```
dn: dc=net
   objectClass: domain
   objectClass: top
   dc: net
```

11. **Run `ldapadd` to add the entry to the directory.**

    For example:

```
% cd /opt/SUNWconn/bin
% ./ldapadd -D cn=admin,o=MyISP,c=US -w secret -f /tmp/net.ldif
```

▼ To Create an Additional Data StoreTo Create an Additional Data Store

Use this procedure if you require more than three top-level domains. This procedure shows you how to create an additional data store so that you can have more than four naming contexts. The new data store is created in the same database directory as the original store, but you are not required to keep them in the same location.

Additional data stores will use a different `rootdn` entity. When you bind to the directory you will need to bind as the `rootdn` appropriate to the data store. To avoid this and have a single identity that can be used to bind and make changes in any data store, create a record in the directory representing an administrative user and give that user permission to make changes in any data store. The next procedure, "To Create a Directory Administrator" explains how to create this user and configure permissions.

Refer to *Sun Directory Services 3.1 Administration Guide* for complete details on creating data stores.

1. **Log in to the machine running the Solaris ISP Server directory service.**

2. **Run** `/opt/SUNWconn/sbin/dsadmintool`**.**

   `dsadmintool` is an X application. Make sure your DISPLAY environment variable is properly set if you are not at the console.

3. **Click *Data Store* in the *Create* menu to display the Create Data Store dialog.**

4. **Enter a suffix for at least one naming context in this data store in the *Datastore Suffix* field.**

   For example, `dc=uk`.

   If you want to add more that one suffix, click the *More Suffixes* button.

5. **Enter a path where the data will be stored in the *DB Directory* field.**

   The original data store is located in `/var/opt/SUNWconn/ldap/dbm` by default. You may also want to store additional data stores in `/var/opt/SUNWconn/ldap`.

6. **You may want to change some of the other settings.**

   The following are the default settings for the original Solaris ISP Server data store:

Read-only Datastore	off
Last-modification attributes	on
Cache size (entries)	5000
Cache size (bytes)	100000

Congestion Limits: critical	200
Congestion Limits: congested	500
Congestion Limits: back to normal	1000

7. **Click *OK* to save your changes and create the data store.**

   The data store should now appear in the *Data Store* list in the `dsadmintool` *Properties* page.

8. **Click *Apply* in the `dsadmintool` console window to apply your changes to the configuration.**

9. **Stop and restart the LDAP service using `dsadmintool` or `/etc/init.d/dsserv`.**

10. **Now add a record representing the root node in the naming context of the new data store.**

    The following example ldif record could be used with `ldapadd` to create a record for `dc=uk`:

```
dn: dc=uk
   objectClass: domain
   objectClass: top
   dc: uk
```

To add this record to the new data store, you will have to bind as the rootdn of the new data store. The new rootdn uses the original rootdn common name, but with the new naming context. For example, if the cn of the original rootdn is `admin`, then to add the above record to our new data store with the `dc=uk` naming context:

```
# cd /opt/SUNWconn/bin
# ./ldapadd -D cn=admin,dc=uk -w secret -f /tmp/net.ldif
```

You should create a single administrator record that can be used to bind to all of your data stores to avoid having to bind with different credentials to each data store. See the next procedure, ”To Create a Directory Administrator“ for more information.

Note - Even if you create an administrative user entry, the first record in any new data store must be created by the rootdn defined for that data store. So you must add the root record for the data store's naming context by binding as the data store rootdn. Subsequent records may be added by any dn with appropriate write privileges.

▼ To Create a Directory AdministratorTo Create a Directory Administrator

This procedure explains how to create a record representing an administrator that can bind to any data store an make modifications. You will create a record in the original data store, then add permissions to the access control list (ACL) file to grant this user permission to modify all entries.

This procedure uses an example of creating a record with the distinguished name "cn=admin,o=myISP,c=US" and a password of "secret." Note that the common name and distinguished name you choose can be identical to the common name and distinguished name of the actual rootdn in your data store.

1. **Create an entry in one of your naming contexts for the user that will act as the administrator.**

   You should place the entry in your original OSI tree; for example, a user whose distinguished name is `cn=admin,o=myISP,c=US`.

   The object class can be any object class that accommodates a password attribute. Using the `top` and `person` object classes is sufficient, but you may want to use other object classes (such as `organizationalPerson`) if you want to store more information in the record's attributes.

   The following ldif file is sufficient to create an administrative user:

```
dn: cn=admin,o=myISP,c=US
   objectClass: top
   objectClass: person
   cn: admin
   sn: admin
   userpassword: secret
   description: rootdn record
```

   You can create the record using `ldapadd` or the Deja tool.

2. **Edit the access control lists (ACLs) for the directory service using either** `dsadmintool` **or by editing the** `dsserv.acl.conf` **file.**

   The configuration file is located in `/etc/opt/SUNWconn/ldap/current` by default.

3. **Add an ACL to allow only the administrative user to modify its own record.**

   For entries that match the dn of the administrative user that you just created, allow “self” write access and everyone (*) no access.

   In the `dsserv.acl.conf` file, the following entry would allow only the user we created above to modify its own record:

```
access to dn="cn=admin,o=myISP,c=us"
        by self write
        by * none
```

   The ACLs are applied in the order in which they appear in the file, so make this the first entry in the file.

4. **Add an ACL that allows the administrative user to modify all records in all naming contexts.**

   This ACL should use a dn pattern to match any distinguished name and explicitly allow write access to the administrative user. Use “`.*=.*`” as the dn pattern to match any dn. Be sure to place this entry at or near the top of the ACLs list so that it is matched before any other entries that might block access.

   In the `dsserv.acl.conf` file, the following entry would grant the user we created above permission to modify any record:

```
access to dn=".*=.*"
        by dn="cn=admin,o=myISP,c=us" write
```

5. **Stop and restart the LDAP service using `dsadmintool` or `/etc/init.d/dsserv`.**

Index