Solaris CIFS Administration Guide

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

Part No: 820–2429–05
March 2009

090321@21990

Contents

Preface

The *Solaris CIFS Administration Guide* describes the Solaris™ Common Internet File System (CIFS) service. This book is intended for system administrators and end users. Both Solaris Operating System (Solaris OS) and Windows system administrators can use this information to configure and integrate the Solaris CIFS service into a Windows environment. In addition, system administrators can configure the identity mapping service. Finally, the chapter about the Solaris CIFS client is primarily intended for Solaris users who would like to mount CIFS shares. The Solaris CIFS client chapter also includes tasks to be performed by a system administrator.

Note – This Solaris release supports systems that use the SPARC® and x86 families of processor architectures: UltraSPARC®, SPARC64, AMD64, Pentium, and Xeon EM64T. The supported systems appear in the *Solaris OS: Hardware Compatibility Lists* at `http://www.sun.com/bigadmin/hcl`. This document cites any implementation differences between the platform types.

In this document these x86 related terms mean the following:

- "x86" refers to the larger family of 64-bit and 32-bit x86 compatible products.
- "x64" points out specific 64-bit information about AMD64 or EM64T systems.
- "32-bit x86" points out specific 32-bit information about x86 based systems.

For supported systems, see the *Solaris OS: Hardware Compatibility Lists*.

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Typographic Conventions

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

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX® system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

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

◆ ◆ ◆ CHAPTER 1

Windows Interoperability (Overview)

This administration guide provides the information needed to integrate a Solaris™ Common Internet File System (CIFS) server into an existing Windows environment and also describes the Solaris CIFS client, which enables you to mount CIFS shares on Solaris systems.

Windows clients can access CIFS shares from the Solaris CIFS service as if they were made available from a Windows server. This guide focuses only on the information required to integrate the Solaris CIFS service and how to use the Solaris CIFS client. Windows topics are only covered when those topics affect the integration of the Solaris CIFS service into the Windows environment.

This chapter covers the following topics:

- "The Solaris CIFS Service" on page 12
- "Configuring the Solaris CIFS Service – Process Overview" on page 16
- "Utilities and Files Associated With the Solaris CIFS Server and Client" on page 17
- "Authentication, Directory, Naming, and Time Services" on page 22
- "CIFS Shares" on page 22
- "Local CIFS Groups" on page 26

Note – The *Common Internet File System (CIFS)* is an enhanced version of the *Server Message Block (SMB)* protocol, which allows CIFS clients to access files and resources on CIFS servers. The terms SMB and CIFS can be considered interchangeable.

Up-to-date troubleshooting information is available from the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

For information about installing the Solaris CIFS service packages, see Getting Started With the Solaris CIFS Service wiki on the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

The Solaris CIFS Service

The Solaris Operating System (Solaris OS) has reached a new level of Windows interoperability with the introduction of an integrated *CIFS service*. A Solaris server can now be an active participant in a Windows active directory domain and provide ubiquitous, cross-protocol file sharing through CIFS and NFS to clients in their native dialect.

The Solaris CIFS service allows a native Solaris system to serve files, by means of CIFS *shares*, to CIFS/SMB enabled clients, such as Windows and Mac OS systems. By virtue of the Solaris CIFS service, a Windows client (or other CIFS client) can interoperate with the Solaris CIFS service as it would with a Windows server.

The Solaris CIFS service can operate in either workgroup mode or in domain mode. In workgroup mode, the Solaris CIFS service is responsible for authenticating users locally when access is requested to shared resources. This authentication process is referred to as local login. In domain mode, the Solaris CIFS service uses pass-through authentication, in which user authentication is delegated to a domain controller.

When a user is successfully authenticated, the Solaris CIFS service generates an access token using the security identifiers (SIDs) that represent the user's identity and the groups of which the user is a member. When the user requests access to files or resources from the service, the access token is used to determine access to files by cross-checking the token with the access control list (ACL) or permissions on files and resources. Solaris OS credentials have been enhanced to fully support Windows-style SIDs. In addition, file systems, such as the ZFS™ file system, support Windows-style ACLs and access checking.

The Solaris OS is unique in that it can manage user identities simultaneously by using both traditional UIDs (and GIDs) and SIDs. When a user is authenticated through the CIFS service, the user's CIFS identity is mapped to the appropriate UNIX® or *Network Information Service (NIS)* identity by using the `idmap` identity mapping service. If an existing UNIX or NIS identity exists, that identity is used. Otherwise, a temporary identity is generated using ephemeral UIDs and GIDs, as required. Ephemeral IDs are valid only within each Solaris OS instance and only until the system is rebooted. These IDs are never stored on disk or transmitted over the network. When an ACL is stored on disk through the CIFS service, the SIDs are used to generate the access control entries. Solaris utilities, such as `ls` and `chmod`, support ACL management.

For more information about how the Solaris OS manages user identities, see Chapter 2, "Identity Mapping Administration (Tasks)."

The following diagram shows how a Solaris file server can operate simultaneously with both NIS and Windows domains. The Windows domain controller provides CIFS authentication and naming services for CIFS clients and servers, while the NIS servers provide naming services for NFS clients and servers.

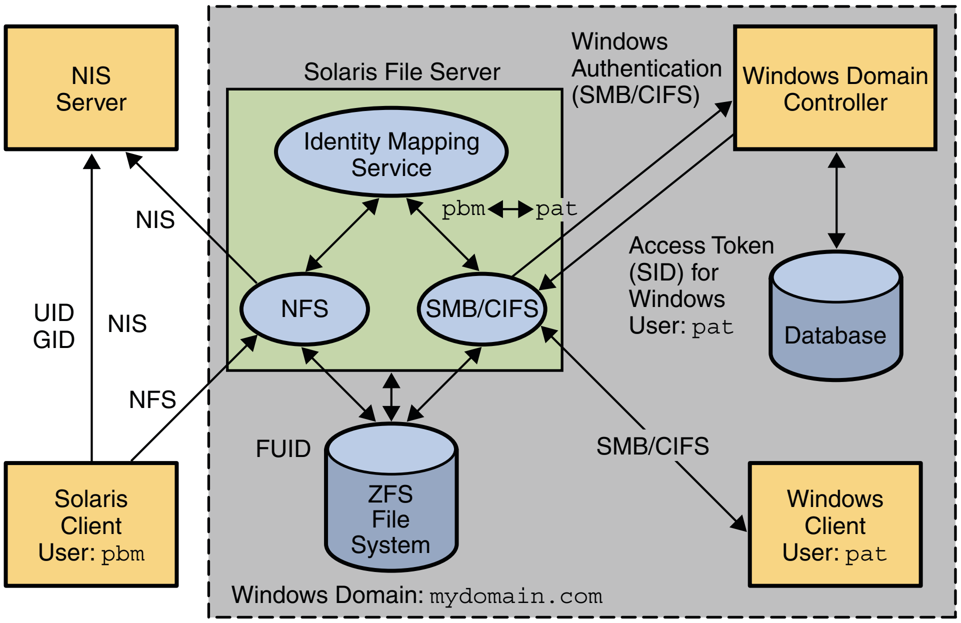

FIGURE 1–1 Solaris CIFS Environment

The Solaris services described in this book include the following components:

- "Solaris CIFS Service" on page 13
- "Solaris CIFS Client" on page 14
- "Identity Mapping Service" on page 15

Solaris CIFS Service

Note – The *Samba* and CIFS services cannot be used simultaneously on a single Solaris system. The Samba service must be disabled in order to run the Solaris CIFS service. For more information, see "How to Disable the Samba Service" on page 83.

For a high-level overview of configuring the Solaris CIFS service, see "Configuring the Solaris CIFS Service – Process Overview" on page 16. For information about configuring the service, see Chapter 3, "Solaris CIFS Service Administration (Tasks)." For more information about the Solaris CIFS service, see the `smbadm`(1M), `smbd`(1M), `smbstat`(1M), `smb`(4), `smbautohome`(4), and `pam_smb_passwd`(5) man pages.

The CIFS features offered by the Solaris service depend on the file system being shared. To fully support the Solaris CIFS service, a file system should support the following features:

- If the file system supports the `archive`, `hidden`, `read-only`, and `system` attributes, these attributes are made available as the DOS attributes available on Windows systems. The ZFS file system supports these attributes.
- If the file system supports Solaris extended attributes, they are made available as NTFS alternate data streams.
- The case-sensitivity capabilities of the file system are made available to CIFS clients. To support both Windows-style access and POSIX access, a file system should support mixed-mode, which is simultaneous support for case-sensitive and case-insensitive name operations.

  The Solaris OS supports both the NFS and CIFS protocols, which have different expectations regarding case behavior. For instance, Windows clients typically expect case-insensitive behavior while local applications and NFS clients typically expect case-sensitive behavior. The ZFS file system supports three case modes: case-sensitive, case-insensitive, and mixed. The ZFS file system can indicate case conflicts when in mixed mode. Mixed mode is recommended for maximum multi-protocol compatibility.
- To provide full Windows *access control list (ACL)* support, file systems should be able to store SIDs and they should at least support NFSv4 ACLs.

For information about the supported features of the UFS and ZFS file systems, see the `ufs`(7FS) man page and the *Solaris ZFS Administration Guide*, respectively.

For information about how to access CIFS shares from your client, refer to the client documentation.

Solaris CIFS Client

The SMB protocol is the natural file-sharing protocol used by Windows and Mac OS systems. Samba implements the SMB protocol for UNIX and Linux systems. The Solaris CIFS client is a Solaris virtual file system that provides access to files and directories from the CIFS service.

By using the Solaris CIFS client, a user can mount remote CIFS shares (directories) on his Solaris system to get read-write access to previously inaccessible files. The Solaris CIFS client does not include the ability to print by means of CIFS or the ability to access CIFS resources other than files and directories. The Solaris CIFS client enables an unprivileged user to mount and unmount shares on directories he owns.

For more information about how to use the Solaris CIFS client to access shares, see Chapter 4, “Solaris CIFS Client Administration (Tasks),” and the `smbutil(1)`, `mount_smbfs(1M)`, `nsmbrc(4)`, `pam_smbfs_login(5)`, and `smbfs(7FS)` man pages.

Identity Mapping Service

The Solaris OS includes an identity mapping service that enables you to map identities between Solaris systems and Windows systems.

This identity mapping service supports the following types of mappings between Windows security identities (SIDs) and Solaris user IDs and group IDs (UIDs and GIDs):

- **Name-based mapping.** Maps Windows and Solaris users and groups by name in the following ways:
  - **Directory-based mapping.** Uses name mapping information that is stored in user or group objects in the Active Directory (AD) and/or the native LDAP directory service to map users and groups.
  - **Rule-based mapping.** An administrator uses rules to map Windows and Solaris users and groups by name.
- **Ephemeral ID mapping.** A UID or GID is dynamically allocated as needed for every SID that is not already mapped by name. Ephemeral ID mapping is used by default.
- **Local SID mapping.** A non-ephemeral UID or GID is mapped to an algorithmically generated local SID.

The `idmap` utility can be used to create and manage the name-based mappings and to monitor the mappings in effect.

For more information about mapping user and group identities, see “Mapping User and Group Identities” on page 29. For information about how to determine your identity mapping strategy, see “Creating Your Identity Mapping Strategy” on page 34. For instructions on how to use the `idmap` command, see “Managing Directory-Based Identity Mapping for Users and Groups (Task Map)” on page 37, “Managing Rule-Based Identity Mapping for Users and Groups (Task Map)” on page 49, and the `idmap(1M)` man page.

Managing Solaris CIFS Configuration Properties

The Solaris CIFS service and the Solaris CIFS client use the `sharectl` command to manage configuration properties. For descriptions of the Solaris CIFS service properties, see the `sharectl(1M)` and `smb(4)` man pages. For descriptions of the Solaris CIFS client properties, see the `nsmbrc(4)` man page.

The Solaris CIFS properties and their values are stored in the Service Management Facility (SMF). For more information about SMF, see Chapter 16, “Managing Services (Overview),” in *System Administration Guide: Basic Administration*.

The `sharectl` command is used throughout the configuration process to set and view properties. This command and examples of its use are described in Chapter 3, "Solaris CIFS Service Administration (Tasks)." The `sharectl` command is also used by the Solaris CIFS client to configure the global environment. For more information, see Chapter 4, "Solaris CIFS Client Administration (Tasks)."

Configuring the Solaris CIFS Service – Process Overview

This section describes the high-level process for configuring the Solaris CIFS service.

1. Determine your identity mapping strategy.

   See "Creating Your Identity Mapping Strategy" on page 34.
2. Configure the Solaris CIFS service as a client to the various services that are used in your environment.
   - For WINS, see "How to Configure WINS" on page 62.

   Your Solaris system might need to be a client of other services that are available in your environment.

   The following list shows these other services and points to information about how to configure your Solaris system as a client of that service.
   - For DNS, see *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*.
   - For Kerberos, see "Configuring Kerberos Clients (Task Map)" in *System Administration Guide: Security Services*.
   - For LDAP, see Chapter 12, "Setting Up LDAP Clients (Tasks)," in *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*.
   - For NIS, see "Setting Up NIS Clients" in *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*.
   - For NTP, see "How to Set Up an NTP Client" in *System Administration Guide: Network Services*.
3. Determine whether you want the Solaris CIFS service to join an existing *Windows domain* or a *Windows workgroup*.
   - To join a domain, see "How to Configure the Solaris CIFS Service in Domain Mode" on page 63.
   - To join a workgroup, see "How to Configure the Solaris CIFS Service in Workgroup Mode" on page 65.
4. Define one or more CIFS shares.

   See "Managing CIFS Shares (Task Map)" on page 67.

Utilities and Files Associated With the Solaris CIFS Server and Client

This section describes the CIFS utilities and files that are used by the CIFS service and client.

- "Solaris CIFS Utilities" on page 17
- "Solaris CIFS Service Daemon" on page 20
- "Solaris CIFS Files" on page 21

Note – The Solaris CIFS service is only supported in the global zone.

Solaris CIFS Utilities

These utilities must be run as superuser or with specific privileges to be fully effective, but requests for information can be made by all users:

- "`mount_smbfs` Command" on page 17
- "`sharectl` Command" on page 18
- "`sharemgr` Command" on page 18
- "`smbadm` Command" on page 19
- "`smbstat` Command" on page 19
- "`smbutil` Command" on page 19
- "`umount_smbfs` Command" on page 20

`mount_smbfs` Command

With this command, you can attach a named CIFS share to a specified mount point. The `mount_smbfs` command enables you to mount a CIFS share to a directory you own without having to become superuser.

For more information, see the following:

- "How to Mount a CIFS Share on a Directory You Own" on page 88
- "How to Mount a Multiuser CIFS Share" on page 94
- "How to Add an Automounter Entry for a CIFS Share" on page 96

Also, see the `mount_smbfs`(1M) man page.

`sharectl` Command

The `sharectl` utility is an administrative tool that enables you to configure and manage file-sharing protocols, such as CIFS and NFS, and network protocols such as NetBIOS. You can use this command to do the following:

- Set client and server operational properties
- Display property values for a specific protocol
- Obtain the status of a protocol

For procedures that use the `sharectl` utility, see the following:

- "How to Configure WINS" on page 62
- "How to Customize the Global Solaris CIFS Environment" on page 95
- "How to View the Global Solaris CIFS Environment Property Settings" on page 96

Also, see the `sharectl(1M)` man page.

`sharemgr` Command

The `sharemgr` utility is an administrative tool that provides an enhanced method of sharing files and performing related tasks. The `sharemgr` utility introduces the following concepts:

- **Share.** One or more files or directories in a share group.
- **Share group.** A container of one or more shared files or directories.

  Note the following:

  - Options for `sharemgr` are set to a share group, not to a specific file or directory. All options apply to each file and directory in the group.
  - A file or directory can only be assigned to one share group. However, you can move a file or directory from one group to another.
  - A share group can be used by multiple file system types. For example, the share group `my_group` could be used by the NFS and ZFS file systems and be assigned one set of options for NFS and another set of options for the ZFS file system.

  Note – When a share is managed by the ZFS file system, `sharemgr` identifies the share and lists it in a `zfs` share group.

Note – The `sharemgr` utility provides a unique way of checking the validity of a desired configuration. The `-n` option allows you to test the validity of the options and properties you want to use with a specific subcommand. The test does not change your configuration. For example, if you use the `-n` option with the subcommand `create`, no share group is created.

You can also use the ZFS file system `sharesmb` property to configure SMB sharing. For more information, see "How to Create a CIFS Share (`zfs`)" on page 69 and the `zfs`(1M) and `zpool`(1M) man pages.

For procedures that use the `sharemgr` utility, see the following:

- "How to Create a CIFS Share (`sharemgr`)" on page 74
- "How to Modify CIFS Share Properties (`sharemgr`)" on page 75
- "How to Remove a CIFS Share (`sharemgr`)" on page 76

Also, see the `sharemgr`(1M) man page.

`smbadm` Command

You can use the `smbadm` command to manage domain membership of the Solaris CIFS service. For instance, you can have the Solaris CIFS service use domain mode or workgroup mode. The `smbadm` command also enables you to configure CIFS local groups. CIFS local groups can be used when Windows accounts must be members of some local groups and when Windows-style privileges must be granted. Solaris local groups cannot provide these functionalities.

For procedures that use the `smbadm` utility, see the following:

- "How to Configure the Solaris CIFS Service in Domain Mode" on page 63
- "How to Configure the Solaris CIFS Service in Workgroup Mode" on page 65
- "How to Create a CIFS Group" on page 80
- "How to Add a Member to a CIFS Group" on page 81
- "How to Remove a Member From a CIFS Group" on page 82
- "How to Modify CIFS Group Properties" on page 82

Also, see the `smbadm`(1M) man page.

`smbstat` Command

You can use the `smbstat` command to show statistical information about the `smbd` server. By default, the `smbstat` command shows general information about the CIFS service as well as dispatched CIFS request counters. For more information, see the `smbstat`(1M) man page.

The `kstat` command can be used to report on kernel CIFS statistics on a periodic basis and also to specify information about individual CIFS statistics. For more information, see the `kstat`(1M) man page.

`smbutil` Command

Use the `smbutil` command to perform the following CIFS client tasks:

- View the shares available for mounting from a particular CIFS server
- Generate a hash of a password for storing in a file such as `$HOME/.nsmbrc`

- Create or remove persistent passwords used to authenticate to CIFS servers
- Resolve a name to an IP address for a server that uses CIFS over NetBIOS, not TCP
- Resolve the specified server to the NetBIOS workgroup and system name

For procedures that use the `smbutil` utility, see the following:

- "How to Find Available CIFS Shares on a Known File Server" on page 86
- "How to Mount a CIFS Share on a Directory You Own" on page 88
- "How to Store a CIFS Persistent Password" on page 89
- "How to Configure the PAM Module to Store a CIFS Persistent Password" on page 90
- "How to Delete a CIFS Persistent Password" on page 92
- "How to Mount a Multiuser CIFS Share" on page 94
- "How to Delete All CIFS Persistent Passwords" on page 98

Also, see the `smbutil`(1) man page.

`umount_smbfs` Command

With this command, you can remove a named CIFS share from a mount point.

For more information, see "How to Unmount a CIFS Share From a Directory You Own" on page 89, and the `mount_smbfs`(1M) man page.

Solaris CIFS Service Daemon

The `smbd` daemon supports CIFS activities on Solaris systems. The `smbd` daemon provides the gateway to the various user space components that support non-file I/O CIFS services. Similar to the NFS kernel service, the SMB kernel module provides SMB file I/O services directly between the network interface and the virtual file system (VFS) within the kernel. Whenever a non-file I/O request is received, such as a user authentication or an MS-RPC named pipe request, it is passed to `smbd` for processing in user space. Requests that require interaction with a domain controller are passed to the SMB Redirector, which provides a simple user space SMB client for IPC communication.

The `smbd` daemon depends on the `idmapd` daemon. For more information about the identity mapping service, see Chapter 2, "Identity Mapping Administration (Tasks)," and the `idmap`(1M) and `idmapd`(1M) man pages.

`smbd` is part of the `svc:/network/smb/server:default` service.

For more information, see the `smbd`(1M) man page.

Solaris CIFS Files

The following files support CIFS activities on any computer:

- `/etc/auto_direct`
- `/etc/smbautohome`
- `$HOME/.nsmbrc`

`/etc/auto_direct` File

Use the `/etc/auto_direct` file to automatically mount a CIFS share when a user accesses the mount point. To use the automount feature, you must store a persistent password for authentication to mount the share. See "How to Store a CIFS Persistent Password" on page 89.

For instructions and examples, see "How to Add an Automounter Entry for a CIFS Share" on page 96.

`/etc/smbautohome` File

The `/etc/smbautohome` file is used to define the automatic sharing rules to be applied when a user connects to the Solaris CIFS service. For more information, see "Autohome Shares" on page 23 and the `smbautohome`(4) man page.

`$HOME/.nsmbrc` File

You can use the `$HOME/.nsmbrc` file to override global behavior of the Solaris CIFS client. Global values are set in the Service Management Facility (SMF). The `.nsmbrc` file is used to customize the behavior of the Solaris CIFS client on a per-user basis.

By default, settings in the `$HOME/.nsmbrc` file are used unless they have security implications, in which case the stronger security setting is used.

For procedures that refer to the `$HOME/.nsmbrc` file, see the following:

- "How to Customize Your Solaris CIFS Environment" on page 93
- "How to Customize the Global Solaris CIFS Environment" on page 95
- "How to View the Global Solaris CIFS Environment Property Settings" on page 96

Also, see the `nsmbrc`(4) man page.

Authentication, Directory, Naming, and Time Services

This section describes the various services that the Solaris CIFS service interoperates with as a client.

The Solaris CIFS service interoperates with a variety of naming services that are used by Windows and Solaris system networks. These naming services include the following:

- **Active Directory Service (AD).** AD is a Windows 2000 directory service that is integrated with the *Domain Name System (DNS)*. AD runs only on domain controllers. In addition to storing and making data available, AD protects network objects from unauthorized access and replicates objects across a network so that data is not lost if one domain controller fails.
- **Domain Name System (DNS).** DNS resolves host names to Internet Protocol (IP) addresses for the system. This service enables you to identify a server by either its IP address or its name.
- **Dynamic DNS (DDNS).** DDNS is provided with AD and enables a client to dynamically update its entries in the DNS database.
- **Lightweight Data Access Protocol (LDAP).** LDAP is a standard, extensible directory access protocol that enables clients and servers that use LDAP naming services to communicate with each other.
- **Network Information Service (NIS).** NIS is a naming service that focuses on making network administration more manageable by providing centralized control over a variety of network information. NIS stores information about the network, machine names and addresses, users, and network services.
- **Network Time Protocol (NTP).** NTP is a protocol that enables a client to automatically synchronize its system clock with a time server. The clock is synchronized each time the client is booted and any time it contacts the time server.
- **Windows Internet Naming Service (WINS).** A *WINS* server resolves *NetBIOS names* to IP addresses, which allows computers on your network to locate other NetBIOS devices more quickly and efficiently. The WINS server runs on a Windows system. The WINS server performs a similar function for Windows environments as a DNS server does for UNIX environments. For more information, see "How to Configure WINS" on page 62.

CIFS Shares

A shared resource, or *share*, is a local resource on a server that is accessible to CIFS clients on the network. For the Solaris CIFS service, a share is typically a directory. Each share is identified by a name on the network. A CIFS client sees the share as a complete entity on the CIFS server, and does not see the local directory path to the share on the server.

Note – A share and a directory are independent entities. Removing a share does not affect the underlying directory.

Shares are commonly used to provide network access to home directories on a network file server. Each user is assigned a home directory. A share is persistent and remains defined regardless of whether users are connected to the server.

The Solaris CIFS service provides a special kind of share called an autohome CIFS share. An *autohome share* is a transient share of a user's home directory that is created when a user logs in and removed when the user logs out.

When a user browses the system, only statically defined shares and his autohome share will be listed.

Autohome Shares

The autohome share feature eliminates the administrative task of defining and maintaining home directory shares for each user that accesses the system through the SMB protocol. The system creates autohome shares when a user logs in, and removes them when the user logs out. This process reduces the administrative effort needed to maintain user accounts, and increases the efficiency of service resources.

For example, if `/home` is a home directory that contains subdirectories for users `bob` and `sally`, you can manually define the shares as follows:

`bob`	`/home/bob`
`sally`	`/home/sally`

However, defining and maintaining directory shares in this way for each user is inconvenient. Instead, you can use the autohome feature.

Note – The Solaris CIFS client does not support autohome shares.

To configure the autohome feature, you need to specify autohome share rules. For example, if a user's home directory is `/fort/sally`, the autohome path is `/fort`. The temporary share is named `sally`. Note that the user's home directory name must be the same as the user's login name. See "How to Create a Specific Autohome Share Rule" on page 76.

When a user logs in, the Solaris CIFS service looks for a subdirectory that matches the user's name based on any rules that have been specified. If the service finds a match and if that share does not already exist, the subdirectory is added as a transient share. When the user logs out, the service removes that transient share.

Some Windows clients log a user out after 15 minutes of inactivity, which results in the autohome share disappearing from the list of defined shares. This behavior is expected for CIFS autohome shares. Even after a CIFS autohome share is removed, the share reappears when the user attempts to access the system (for example, in an Explorer window).

Note – All autohome shares are removed when the Solaris CIFS service is restarted.

Autohome Entries

The Solaris CIFS service can automatically share home directories when a CIFS client connects. The autohome map file, `/etc/smbautohome`, uses the search options and rules to determine whether to share a home directory when a CIFS client connects to the service.

For example, the following entries specify the autohome rules for a particular environment:

```
+nsswitch        dn=ads,dn=sun,dn=com,ou=users
jane    /home/?/&    dn=ads,dn=sun,dn=com,ou=users
```

The `nsswitch` autohome entry uses the naming service to match users to home directories. The second autohome entry specifies that the home directory for user `jane` is `/home/j/jane`.

Autohome Map Entry Format

A map entry, also referred to as a mapping, uses the following format:

key location `[` *container* `]`

key is a user name, *location* is the fully qualified path for the user's home directory, and *container* is an optional AD container.

If you intend to publish the share in AD, you *must* specify an AD container name, which is specified as a comma-separated list of attribute name-value pairs. The attributes use the *Lightweight Data Access Protocol (LDAP)* distinguished name (DN) or relative distinguished name (RDN) format.

The DN or RDN must be specified in LDAP format by using the following attribute types:

- `cn=` represents the common name
- `ou=` represents the organizational unit
- `dc=` represents the domain component

Note – The attribute type that is used to describe an object's RDN is called a *naming attribute*.

AD uses the naming attributes as follows:

- `cn` for the `user` object class
- `ou` for the `OU` (organizational unit) object class
- `dc` for the `domainDns` object class

Autohome Map Key Substitution

The autohome feature supports the following wildcard substitutions for the value of the key field:

- The ampersand character (&) is expanded to the value of the key field for the entry in which it occurs. In the following example, `&` expands to `jane`:

  ```
  jane /home/&
  ```

- The question mark character (`?`) is expanded to the value of the first character in the key field for the entry in which it occurs. In the following example, the path is expanded to `/home/jj/jane`:

  ```
  jane /home/??/&
  ```

Wildcard Rule

When supplied in the key field, the asterisk character (`*`) is recognized as the "catch-all" entry. Such an entry matches any key not previously matched.

For example, the following entry would map any user to a home directory in `/home` in which the home directory name was the same as the user name:

```
*     /home/&
```

Note – The wildcard rule is *only* applied if an appropriate rule is not matched by another map entry.

`nsswitch` Map

The `nsswitch` map is used to request that the home directory be obtained from a password database, such as the local, NIS, or LDAP database. If an AD path is appended, it is used to publish shares.

```
+nsswitch
```

Like the "catch-all" entry, the `nsswitch` map is *only* searched if an appropriate rule is not matched by another map entry.

Note – The wildcard and `nsswitch` rules are mutually exclusive. Do not include an `nsswitch` rule if a wildcard rule has already been defined.

Local CIFS Groups

Local CIFS groups can be created on the system that runs the Solaris CIFS service. These CIFS groups apply only to users that are connected through CIFS.

The Solaris CIFS service supports the following built-in CIFS groups:

- **Administrators.** Members of this group can fully administer files and directories on the system.
- **Backup Operators.** Members of this group can bypass file security to back up and restore files.
- **Power Users.** Members of this group can be assigned ownership of files and directories on the system, and can back up and restore files.

Local groups use privileges to provide a secure mechanism for assigning task responsibility on a system-wide basis. Each privilege has a well-defined role assigned by the system administrator to a user or a group.

Unlike access rights (which are assigned as permissions on a per-object basis through security descriptors), privileges are independent of objects. Privileges bypass object-based access control lists to allow the holder of the privilege to perform the role assigned. For example, members of the Backup Operators group must be able to bypass normal security checks to back up and restore files they would normally not be able to access.

The following definitions show the difference between an access right and a privilege:

- An *access right* is explicitly granted or denied to a user or a group. Access rights are assigned as permissions in a discretionary access control list (DACL) on a per-object basis.
- A *privilege* is a system-wide role that implicitly grants members of a group the ability to perform predefined operations. Privileges override or bypass object-level access rights.

You can assign any of the privileges to any of the local groups. Because you can make any domain user a member of the local groups, you can assign these privileges to any domain user.

The following privileges are supported for local groups:

- **Back up files and directories.** Perform backups without requiring read access permission on the target files and folders.
- **Restore files and directories.** Restore files without requiring write access permission on the target files and folders.
- **Take ownership of files and folders.** Take ownership of an object without requiring take-ownership access permission. Ownership can only be set to those values that the holder of the privilege may legitimately assign to an object.

By default, members of the local Administrators group can take ownership of any file or folder, and members of the Backup Operators group can perform backup and restore operations. Members of the Power Users group do not have default privileges.

For information about managing CIFS groups, see "Managing CIFS Groups (Task Map)" on page 79.

CHAPTER 2

Identity Mapping Administration (Tasks)

This chapter describes the identity mapping service that maps Windows security identifiers (SIDs) to Solaris user identifiers (UIDs) and group identifiers (GIDs). The chapter also includes instructions on how to manage name-based mappings.

This chapter covers the following topics:

The `idmapd` service can run in the global zone or in non-global zones. However, if Solaris Trusted Extensions software is enabled, the `idmapd` service *must* run in the global zone.

Note – CIFS is an enhanced version of the SMB protocol, which allows CIFS clients to access files and resources on CIFS servers. The terms SMB and CIFS can be considered interchangeable.

Up-to-date troubleshooting information is available from the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

Mapping User and Group Identities

The Solaris CIFS service is designed to reside in a multiprotocol environment and provide an integrated model for sharing data between Windows and Solaris systems. Although files can be accessed simultaneously from both Windows and Solaris systems, no industry-standard mechanism is used to define a user in both Windows and Solaris environments. Objects can be created in either environment, but traditionally the access control semantics for each environment are vastly different. The Solaris OS is adopting the Windows model of access control lists (ACLs) by introducing ACLs in NFSv4 and the ZFS file system, and by providing the `idmapd` identity mapping service.

The Solaris CIFS service uses identity mapping to establish an equivalence relationship between a Solaris user or group and a Windows user or group in which both the Solaris and Windows identities are deemed to have equivalent rights on the system.

The Solaris CIFS service determines the Windows user's Solaris credentials by using the `idmapd` service to map the SIDs in the user's Windows access token to UIDs and GIDs, as appropriate. The service checks the mappings and if a match for the Windows domain name and Windows entity name is found, the Solaris UID or GID is taken from the matching entry. If no match is found, an ephemeral UID or GID is dynamically allocated. An *ephemeral ID* is a dynamic UID or GID mapping for an SID that is not already mapped by name. An ephemeral ID does not persist across Solaris system reboots. Ephemeral mappings enable the Solaris CIFS service to work in a Windows environment without having to configure any name-based mappings.

The `idmapd` service supports the following types of mappings between Windows security identifiers (SIDs) and Solaris user IDs and group IDs (UIDs and GIDs):

- **Name-based mapping.** Maps Windows and Solaris users and groups by name when the Solaris CIFS service is in domain mode. For more information, see "The Solaris CIFS Service" on page 12. Name-based mapping works in the following ways:
  - **Directory-based mapping.** If configured, `idmapd` first tries to use name mapping information that is stored in user or group objects in the Active Directory (AD), in the native LDAP directory service, or in both. For instance, an AD object for a particular Windows user or group can be augmented to include the corresponding Solaris user or group name. Similarly, the native LDAP object for a particular Solaris user or group can be augmented to include the corresponding Windows user or group name.

    You can configure `idmapd` to use AD and/or native LDAP directory-based name mappings by setting the `idmap` service properties in SMF. See Service Properties in the `idmap`(1M) man page.

    If directory-based name mapping is not configured or if it is configured but not found, `idmapd` will process locally stored rule-based mappings.
  - **Rule-based mapping.** An administrator maps Windows and Solaris users and groups by name.
- **Ephemeral ID mapping.** A UID or GID is dynamically allocated for every SID that is not mapped by name.
- **Local SID mapping.** A non-ephemeral UID or GID is mapped to an algorithmically generated local SID if it is not mapped by name.

You can use the `idmap` command to create and manage the rule-based mappings.

When you specify rule-based mappings, you must specify the direction in which the mapping occurs, as follows:

- **Bidirectional mapping.** Map the specified Windows name to the specified Solaris name, and map the specified Solaris name to the specified Windows name. By default, rule-based mappings that you create are bidirectional.

  The following example shows a bidirectional mapping of the Windows user `dana@example.com` to `danas`, the Solaris user. Note that `dana@example.com` maps to `danas`, and `danas` maps to `dana@example.com`.

  ```
  dana@example.com == danas
  ```
- **Unidirectional mapping.** Map the names only in the specified direction.

  The following example combines unidirectional and bidirectional mappings to map between Windows users `dana@example.com` and `danasan@example.com` and Solaris user `danas`. The bidirectional rule maps between Windows user `dana@example.com` and Solaris user `danas`. The unidirectional rule maps Windows user `danasan@example.com` to the Solaris user `danas`. When Solaris user `danas` needs to map to the appropriate Windows user, it maps to `dana@example.com`.

  ```
  dana@example.com == danas
  danasan@example.com => danas
  ```

On Windows and Solaris systems, files have an owner attribute and a group attribute. A Solaris file owner attribute must be a UID, and the group attribute must be a GID. Unlike the Solaris OS, Windows has no such restrictions. Windows permits either a user SID or a group SID to be a file owner or a file group. In fact, Windows uses the Administrator Group as a file owner in many instances, and any Windows application can set the file owner and group attributes to any SID.

The Solaris system cannot interchange UIDs and GIDs like Windows can. Therefore, the Solaris system must be able to perform the following types of mappings:

- Map a group SID to a UID when the group SID occurs in an owner field
- Map a user SID to a GID when the user SID occurs in group field

These are called *diagonal mappings*, which use naming rules to set up the mappings.

Solaris Users and Groups

Solaris users and groups can be defined in local files (`/etc/passwd` and `/etc/group`) or in a naming or directory service, such as NIS and LDAP. The naming services you configure are listed in the Solaris naming services switch file `/etc/nsswitch.conf`. For more information, see Chapter 2, "The Name Service Switch (Overview)," in *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*.

The Solaris CIFS service can be configured as a client of the various distributed naming services, such as NIS and LDAP. For information about configuring the Solaris CIFS service as a client for these naming services, see *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*.

Each user and group is assigned a 32-bit identifier known, respectively, as a *user ID (UID)* and a *group ID (GID)*. The Solaris OS has extended the `uid_t` and `gid_t` types from signed to unsigned 32-bit integers. Now that the `uid_t` and `gid_t` types are unsigned, the upper half of these namespaces is available for ephemeral dynamic ID mapping. This mapping process enable IDs to be assigned dynamically and ephemerally on demand. An *ephemeral mapping* is one that does not survive a Solaris system reboot. Typically, the UID or GID uniquely identifies a user or group within a single Solaris domain. However, these values are not unique across domains.

Traditionally, UID 0 or GID 0 is assigned to the `root` user or group. The `root` user is granted almost unlimited access to system objects in order to perform administration tasks.

Windows Users and Groups

Windows users and groups are defined in a *Security Account Manager (SAM) database*, which is managed on a *Windows domain controller*. Each user and group is identified by a security identifier (SID). An *SID* is a variable-length structure that uniquely identifies a user or group both within a host and a local domain, and across all possible Windows domains.

The text form of an SID is represented as follows:

```
S-R-I-SA-SA-...-SA
```

The following describes the fields in the SID text string:

- `S` – Identifies the string as an SID.
- *R* – Identifies the revision number, which is currently 1.
- *I* – Identifies the 48-bit identifier authority value, which is the agent or namespace that issued the SID.
- *SA* – Is one or more subauthorities, such as a *relative identifier (RID)*.

In a domain SID, the RIDs identify the domain. In a user or group SID, except for the last RID, the RIDs identify the machine or the domain that issues the SID. The last RID identifies the user or group.

For example, the `S-1-5-32-500` SID contains a version number of 1. The identifier authority value is 5, and it contains the 32 and 500 subauthorities. The value 500 is the RID.

The `idmapd` service generates a unique SID for the host on which it runs. This SID is used to represent both users and groups that cannot be mapped by name to SIDs. This SID is stored in the equivalent of a local SAM database. The Solaris computer SID is generated randomly.

The `idmap` service generates a unique SID, *machine-SID*, for the host on which it runs. This SID is used to generate local SIDs as follows:

```
local SID for user = machine-SID - 1000 + user's-UID
local SID for group = machine-SID - 2^31 + group's-GID
```

For instance, the local SID for a user with a UID of `182048` and a machine SID of `S-1-5-21-726303253-4128413635` is `S-1-5-21-726303253-4128413635-183048`.

Local SIDs are used to represent Solaris users or groups that have non-ephemeral UIDs or GIDs and that cannot be mapped by name.

Configuring DNS for Identity Mapping in Domain Mode

The `idmap` service requires that DNS be configured properly before you join a Solaris system to an AD domain.

Note – This DNS configuration is only required for domain mode though the `idmap` service operates in workgroup mode as well. When in workgroup mode, domain name-based mapping is not performed.

The `idmapd` daemon uses DNS information that is specified in the `/etc/resolv.conf` configuration file to discover its domain.

The domain is specified by the value of the `domain` or `search` configuration directive.

- `domain` – The value specifies the domain to use for auto-discovery. The following example specifies the `sales.example.com` domain:

  ```
  domain sales.example.com
  ```
- `search` – The first domain name listed specifies the domain to use for auto-discovery. The following example specifies the `sales.example.com` domain:

  ```
  search sales.example.com example.com
  ```

If both the `domain` and `search` directives are used, the last directive that is specified determines the domain to be used for auto-discovery.

The `idmapd` daemon discovers the domain controller and the global catalog by performing DNS lookups for SRV records. These SRV records are generated by the DNS server that is part of AD on the Windows domain controller. Therefore, the simplest way to configure DNS is to point to the DNS server on the Windows domain controller.

The `idmap` service looks for the following SRV records:

```
_ldap._tcp.dc._msdcs.domain-name
_ldap._tcp.site-name._sites.dc._msdcs.domain-name
_ldap._tcp.gc._msdcs.forest-name
_ldap._tcp.site-name._sites.gc._msdcs.forest-name
```

You can verify that the configuration is working properly by running the following, which should return the name of the Windows domain controller:

```
# dig _ldap._tcp.domain-name SRV +short
```

For example, the following returns the domain controller for the `sales.example.com` domain:

```
# dig _ldap._tcp.sales.example.com SRV +short
0 100 389 test-win2k3.sales.example.com.
```

After DNS is correctly configured, you can join the Solaris system to an AD domain by using the `smbadm` or `kclient` utility. For more information about using the `smbadm` command see, "How to Configure the Solaris CIFS Service in Domain Mode" on page 63 and the `smbadm`(1M) man page. For information about the `kclient` command, see the `kclient`(1M) man page.

Note – If the `idmap` service is unable to discover an AD server, the service only handles mappings for well-known SIDs and local SIDs.

Creating Your Identity Mapping Strategy

Windows SID to Solaris UID and GID mapping is required when the Solaris CIFS service is deployed to a Windows environment. The *identity mapping* enables Windows clients to transparently access CIFS shares and remote services from the Solaris CIFS service.

Your Solaris CIFS service can use name-based mapping, ephemeral ID mapping, or both. By default, the server uses ephemeral ID mapping, which dynamically assigns an ephemeral ID as a UID or a GID for a particular Windows SID. The identity mapping strategy you choose depends on the type of Windows environment you have.

- **Using name-based mapping.** If your Windows environment includes a parallel Solaris naming service infrastructure, such as NIS, you might want to use *name-based mappings* to associate Windows users with Solaris users, and Windows groups with Solaris groups.

  Name-based mappings include directory-based mappings and rule-based mappings. A *directory-based mapping* uses name mapping information that is stored in user or group objects in the Active Directory (AD), in the native LDAP directory service, or both to map users and groups. A *rule-based mapping* uses rules to associate Windows users and groups with equivalent Solaris users and groups by name rather than by identifier.

  To use name-based mapping, do the following:

  1. Choose a Windows domain that is the most natural counterpart to the Solaris naming service domain.
  2. Determine whether to use directory-based or rule-based mappings.

     These name-based mapping types have the following strengths and weaknesses:

     - **Directory-based mappings.** Are stored globally and each mapping is configured individually. However, the setup is rather difficult and time-consuming. This method is more suitable if many CIFS servers are being used in your environment.

       If you decide to use directory-based mappings, use one of the following guidelines to determine which naming services to employ:

       - If you have already deployed AD or native LDAP, use that naming service.
       - If you want one-to-one mappings, choose either AD-only or native LDAP-only modes as follows:
         - If you have few native LDAP domains and do most of your administration in AD, choose AD-only mode
         - Otherwise, choose native LDAP-only mode
       - If you need more flexibility than one-to-one mappings offer, choose mixed mode.

         For example, to map Windows entities to one native LDAP user, group, or both, use mixed mode. Similarly, use mixed mode to map multiple native LDAP users or groups to one Windows entity.

         Alternatively, you can employ directory-based mapping *and* name-based rules.
     - **Rule-based mappings.** Are easy to configure and can be configured with a single wildcard rule. However, the mapping rules are only stored on a particular computer rather than being global. This method is more suitable if only one CIFS server is being used in your environment.

  - **Using directory-based mapping.**

1. Extend the AD schema, the native LDAP schema, or both with new attributes to represent a UNIX user name, a UNIX group name, or a Windows name. Also, populate the AD or native LDAP user and group objects, or both types of objects, with the appropriate attribute and value. See "How to Extend the Active Directory Schema, and User and Group Entries" on page 39 and "How to Extend the Native LDAP Schema, and User and Group Entries" on page 42.

   Note – If you do not want to modify the schema and suitable attributes already exist in either AD or native LDAP, use those attributes.

   ---

2. Use the `svccfg` command to enable directory-based mapping on the Solaris system. Also, inform the `idmap` service about the new AD attributes, the native LDAP attributes, or both types of attributes that are used by the user and group objects. See "How to Configure Directory-Based Mapping" on page 44.

- **Using rule-based mapping.**

  1. Create a bidirectional rule-based mapping to map all users in the Windows domain to users of the same name in the Solaris domain.

     ```
     # idmap add 'winuser:*@example.com' 'unixuser:*'
     # idmap add 'wingroup:*@example.com' 'unixgroup:*'
     ```

     The previous commands map not only user names, but group names. For instance, the first command would map the Windows user called `pat@example.com` to the Solaris user `pat`. The second command would map the Windows group called `staff@example.com` to the Solaris group `staff`.

     Note – You can only have one bidirectional rule-based mapping to map all users in a single Windows domain to all Solaris users in the local Solaris domain.

     ---

  2. Create bidirectional rule-based mappings for users and groups whose Windows names do not exactly match the Solaris names.

     ```
     # idmap add winuser:terry@example.com unixuser:terrym
     ```

     The previous command would map a Windows user called `terry@example.com` to the Solaris user `terrym`.

Caution – Rule-based identity mappings can be used to map Windows users and groups to, for example, the `nobody` Solaris user and group. In some circumstances, such a mapping can lock a user out of the CIFS service.

The mapping works on both a per-user and a per-group basis and for entire Windows domains. Successfully using this type of mapping to lock out users depends on the rule-based mappings being in sync with the actual names in the naming service, such as AD. As a result, this type of mapping might not be a reliable way to lock users out of the CIFS service, and should not be used for that purpose.

This scheme *could* be used to lock out a user if the administrator who maintains the user and group namespace is the *same* administrator who maintains the identity mappings. If not, however, you could get into a situation where one administrator creates the rule to lock the user out and another administrator grants a request to change the user name. In that case, the rule created to lock the user out only applies to his old user name, not to the new name. Thus, the user is no longer locked out of the CIFS service as intended.

To ensure that a user is correctly locked out, lock out the user in the naming services.

For example, creating a bidirectional mapping between the `dana@example.com` and `nobody` users does not prevent user `dana@example.com` from bypassing this attempt to deny him access to the CIFS service. He can simply have his user name changed to something else so that the rule will no longer apply.

- **Using ephemeral ID mapping.** If your Windows environment does not already include a parallel Solaris naming service infrastructure, such as NIS, you do not need to create rule-based identity mappings. Instead, the default identity mapping configuration uses ephemeral IDs to map between Windows SIDs and Solaris UIDs and GIDs.

Managing Directory-Based Identity Mapping for Users and Groups (Task Map)

The following table points to the tasks that you can use to manage directory-based identity mapping for the Solaris CIFS service in a Windows environment.

These tasks use the `idmap(1M)` command to manage identity mapping.

Task	Description	For Instructions
Extend the Active Directory (AD) schema with user and group name attributes.	This procedure describes how to extend the AD schema and populate the user and group objects with UNIX user and group name information.	"How to Extend the Active Directory Schema, and User and Group Entries" on page 39
Extend the native LDAP schema with user and group name attributes.	This procedure describes how to extend the native LDAP schema and populate the user and group objects with Windows user and group name information.	"How to Extend the Native LDAP Schema, and User and Group Entries" on page 42
Configure directory-based name mapping.	Use this procedure to enable directory-based mapping. This procedure also informs the `idmap` service about the new AD schema attributes that are used by the user and group objects.	"How to Configure Directory-Based Mapping" on page 44
Add a directory-based name mapping to a user object.	Use this procedure to add a directory-based name mapping to a user object in AD or native LDAP.	"How to Add a Directory-Based Name Mapping to a User Object" on page 45
Add a directory-based name mapping to a group object.	Use this procedure to add a directory-based name mapping to a group object in AD or native LDAP.	"How to Add a Directory-Based Name Mapping to a Group Object" on page 46
Remove a directory-based name mapping from a user object.	Use this procedure to remove a directory-based name mapping from a user object in AD or native LDAP.	"How to Remove a Directory-Based Name Mapping From a User Object" on page 47
Remove a directory-based name mapping from a group object.	Use this procedure to remove a directory-based name mapping from a group object in AD or native LDAP.	"How to Remove a Directory-Based Name Mapping From a Group Object" on page 48

For more information about user and group identities, see "Mapping User and Group Identities" on page 29.

For more information about how to determine your identity mapping strategy, see "Creating Your Identity Mapping Strategy" on page 34.

Note – In a cluster configuration, changes made to user maps and to group maps on one server are immediately propagated to the other server.

▼ How to Extend the Active Directory Schema, and User and Group Entries

This procedure shows how to extend the AD schema and populate the user and group objects with the associated Solaris names.

Note – Perform this task before enabling directory-based mapping on your Solaris system.

---

1. **(Optional) Extend the AD schema to add the new UNIX user and group attributes.**

   Note – If you do not want to extend the AD schema, you can use an existing AD schema attribute to store UNIX user and group name information. For instance, if you already have schema that is comparable to what is described in Example 2–1, you can use your attributes instead of creating new ones.

   ---

   a. **Create an LDAP Data Interchange Format (LDIF) file to describe the AD schema changes.**

   For sample LDIF file contents, see Example 2–1. Also see *Extending Your Active Directory Schema in Windows Server 2003 R2* and *Step-by-Step Guide to Using Active Directory Schema and Display Specifiers* on the Microsoft technet web site (`http://technet.microsoft.com/`).

   b. **Use the `ldifde` tool to load the schema changes into AD from the Windows server.**

   ```
   C:\> ldifde -v -i -f input-file
   ```

2. **Use the `ldapmodify` command to populate the AD user and group objects with the new attributes and their values.**

   You can use the `idmap set-namemap` command to populate user and group objects. See "How to Add a Directory-Based Name Mapping to a User Object" on page 45 and "How to Add a Directory-Based Name Mapping to a Group Object" on page 46.

   You can also use any of the Windows AD utilities to populate these objects.

   a. **Create an LDIF file to record the updates to the AD user and group objects.**

   See a sample LDIF file in Example 2–2. For more information about the LDIF file format, see RFC 2849 (`http://www.faqs.org/rfcs/rfc2849.html`).

   b. **Use the `kinit` command to obtain a Kerberos ticket-granting ticket (TGT) for a privileged AD principal.**

   This principal will be used by the `ldapmodify` command to update the AD objects described in the file you created in the previous substep.

For example:

```
$ kinit Administrator
Password for Administrator@EXAMPLE.COM:
```

c. **Use the `ldapmodify` command to update the user objects on the AD server.**

```
$ ldapmodify -h AD-server-name -o mech=gssapi -o authzid='' -f input-file
```

Example 2–1 Extending the AD Schema

The following LDIF example file, `ad_namemap_schema.ldif`, describes the AD schema changes:

```
dn: CN=unixUserName, CN=Schema, CN=Configuration, DC=example, DC=com
changetype: add
attributeID: 1.3.6.1.4.1.42.2.27.5.1.60
attributeSyntax: 2.5.5.3
isSingleValued: TRUE
searchFlags: 1
lDAPDisplayName: unixUserName
adminDescription: This attribute contains the object's UNIX username
objectClass: attributeSchema
oMSyntax: 27

dn: CN=unixGroupName, CN=Schema, CN=Configuration, DC=example, DC=com
changetype: add
attributeID: 1.3.6.1.4.1.42.2.27.5.1.61
attributeSyntax: 2.5.5.3
isSingleValued: TRUE
searchFlags: 1
lDAPDisplayName: unixGroupName
adminDescription: This attribute contains the object's UNIX groupname
objectClass: attributeSchema
oMSyntax: 27

dn:
changetype: modify
add: schemaUpdateNow
schemaUpdateNow: 1
-

dn: CN=unixNameInfo, CN=Schema, CN=Configuration, DC=example, DC=com
changetype: add
governsID: 1.3.6.1.4.1.42.2.27.5.2.15
lDAPDisplayName: unixNameInfo
adminDescription: Auxiliary class to store UNIX name info in AD
mayContain: unixUserName
mayContain: unixGroupName
objectClass: classSchema
```

```
objectClassCategory: 3
subClassOf: top
```

Use the `ldifde` tool to load the schema changes into AD from the Windows server:

```
C:\> ldifde -v -i -f ad_namemap_schema.ldif
```

Example 2–2 Populating AD User and Group Objects

The following example has Windows users `terry`, `cal`, and `dana` stored in Active Directory. These Windows users are associated with the Solaris users `tmw`, `crj`, and `dab`, respectively.

This example shows how to add the Solaris user names to the appropriate user objects in AD by using the `ldapmodify` command.

First, create an input file, `updateUsers`, that associates the Windows names with the Solaris names:

```
$ cat updateUsers
dn: CN=Terry Walters,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: tmw

dn: CN=Cal Jamieson,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: crj

dn: CN=Dana Bloom,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: dab
$
```

Next, use the `kinit` command to obtain a TGT for a privileged principal:

```
$ kinit Administrator
Password for Administrator@EXAMPLE.COM:
```

Finally, run the `ldapmodify` command to update the user objects on the AD server, `saturn`:

```
$ ldapmodify -h saturn -o mech=gssapi -o authzid='' -f updateUsers
```

▼ How to Extend the Native LDAP Schema, and User and Group Entries

This procedure shows how to extend the native LDAP schema and populate the user and group objects with the associated Windows names.

Note – Perform this task before enabling directory-based mapping on your Solaris system.

---

1 **(Optional) Extend the native LDAP schema to add the new Windows user and group attributes.**

Note – If you do not want to extend the native LDAP schema, you can use an existing native LDAP schema attribute to store Windows user and group name information. For instance, if you already have schema that is comparable to what is described in Example 2–3, you can use your attributes instead of creating new ones.

---

a. **Create an LDAP Data Interchange Format (LDIF) file to describe the native LDAP schema changes.**

For sample LDIF file contents, see Example 2–3.

b. **Use the `ldapmodify` tool to load the schema changes into native LDAP.**

```
$ ldapmodify -D cn=admin -w p -f input-file
```

2 **Use the `ldapmodify` command to populate the native LDAP user and group objects with the new attributes and their values.**

You can use the `idmap set-namemap` command to populate user and group objects. See "How to Add a Directory-Based Name Mapping to a User Object" on page 45 and "How to Add a Directory-Based Name Mapping to a Group Object" on page 46.

a. **Create an LDIF file to record the updates to the native LDAP user and group objects.**

See a sample LDIF file in Example 2–4. For more information about the LDIF file format, see RFC 2849 (`http://www.faqs.org/rfcs/rfc2849.html`).

b. **Use the `ldapmodify` command to update the user objects on the native LDAP server.**

```
$ ldapmodify -h LDAP-server-name -o mech=gssapi -o authzid='' -f input-file
```

Example 2–3 Extending the Native LDAP Schema

The following LDIF example file, `nldap_namemap_schema.ldif`, describes the native LDAP schema changes:

```
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.42.2.27.5.1.62
   NAME 'winAccountName'
   DESC 'Windows user or group name corresponding to a Unix user or group'
   EQUALITY caseIgnoreMatch
   SUBSTRINGS caseIgnoreSubstringsMatch
   ORDERING caseIgnoreOrderingMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
-
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.42.2.27.5.2.16
   NAME 'winAccount'
   DESC 'Auxiliary class to store Windows name mappings in Unix user/group objects'
   SUP top
   AUXILIARY
   MAY winAccountName )
```

Use the `ldapmodify` tool to load the schema changes into native LDAP:

```
$ ldapmodify -D cn=admin -w - -f f nldap_namemap_schema.ldif
Enter bind password:
modifying entry cn=schema
```

Example 2–4 Populating Native LDAP User and Group Objects

The following example has Solaris users `tmw`, `crj`, and `dab` stored in native LDAP. These Solaris users are associated with the Windows users `terry`, `cal`, and `dana`, respectively.

This example shows how to add the Windows user names to the appropriate user objects in native LDAP by using the `ldapmodify` command.

First, create an input file, `updateUsers`, that associates the Solaris names with the Windows names:

```
$ cat updateUsers
dn: uid=tmw,ou=passwd,dc=example,dc=com
changetype: modify
add: winAccountName
winAccountName: terry@example.com

dn: uid=crj,ou=passwd,dc=example,dc=com
changetype: modify
add: winAccountame
winAccountame: cal@example.com

dn: uid=dab,ou=passwd,dc=example,dc=com
```

```
changetype: modify
add: winAccountame
winAccountame: dana@example.com
$
```

Then, run the `ldapmodify` command to update the user objects on the native LDAP server, `neptune`:

```
$ ldapmodify -h neptune -o mech=gssapi -o authzid='' -f updateUsers
```

▼ How to Configure Directory-Based Mapping

Before You Begin Before you can enable directory-based mapping on your Solaris system, you must extend the AD schema, the native LDAP schema, or both, and populate the user and group objects with the associated Solaris names. See "How to Extend the Active Directory Schema, and User and Group Entries" on page 39 and "How to Extend the Native LDAP Schema, and User and Group Entries" on page 42.

1. **Enable directory-based mapping.**

   ```
   # svccfg -s svc:/system/idmap setprop config/ds_name_mapping_enabled=boolean: true
   ```

2. **Inform the `idmap` service about the new user and group attributes.**

Note – To fully enable directory-based mapping, you *must* specify values for the following properties depending on the directory service or services you plan to use:

- `config/ad_unixuser_attr`
- `config/ad_unixgroup_attr`
- `config/nldap_winname_attr`

These properties do not have default values. If the properties are not set, directory-based mapping is effectively disabled for the corresponding naming service.

In an environment that stores user and group name information in both Active Directory and native LDAP, perform the steps for each naming service.

- **For Active Directory, inform the `idmap` service about the new Active Directory UNIX user and group attributes.**

  ```
  # svccfg -s svc:/system/idmap setprop \
  config/ad_unixuser_attr=astring: attribute-name
  # svccfg -s svc:/system/idmap setprop \
  config/ad_unixgroup_attr=astring: attribute-name
  ```

  attribute-name is the attribute name you choose for the UNIX user or group name to be stored in AD.

For example, the following specifies the `unixGroupName` and `unixUserName` attribute names for the UNIX group and user names, respectively:

```
# svccfg -s svc:/system/idmap setprop \
config/ad_unixgroup_attr=astring: unixGroupName
# svccfg -s svc:/system/idmap setprop \
config/ad_unixuser_attr=astring: unixUserName
```

- **For native LDAP, inform the `idmap` service about the new native LDAP Windows name attribute.**

  ```
  # svccfg -s svc:/system/idmap setprop \
  config/nldap_winname_attr=astring: attribute-name
  ```

  attribute-name is the attribute name you choose for the Windows name to be stored in native LDAP.

  For example, the following specifies the `winAccountName` attribute name for the Windows name:

  ```
  # svccfg -s svc:/system/idmap setprop \
  config/nldap_winname_attr=astring: winAccountName
  ```

▼ How to Add a Directory-Based Name Mapping to a User Object

This procedure shows how to perform the following directory-based name mapping:

- Map a Windows user to a Solaris user by adding the Solaris user name to the AD object for the specified Windows user.
- Map a Solaris user to a Windows user by adding the Windows user name to the native LDAP object for the specified Solaris user.

For more information about the `idmap set-namemap` command and its options, see the `idmap`(1M) man page.

1 **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine whether to augment a user object in AD or in the native LDAP service.**

- **To augment the Windows user object in AD, type:**

  ```
  # idmap set-namemap winuser:username@domain-name unixuser:username
  ```

  For example, the following command maps Windows user `danab@example.com` to Solaris user `dana` by adding the Solaris name to the AD object for `danab@example.com`:

  ```
  # idmap set-namemap winuser:danab@example.com unixuser:dana
  ```

- **To augment the Solaris user object in native LDAP, type:**

  ```
  # idmap set-namemap unixuser:username winuser:username@domain-name
  ```

  For example, the following command maps Solaris user `dana` to Windows user `danab@example.com` by adding the Windows name to the native LDAP object for `dana`:

  ```
  # idmap set-namemap unixuser:dana winuser:danab@example.com
  ```

▼ How to Add a Directory-Based Name Mapping to a Group Object

This procedure shows how to perform the following directory-based name mapping:

- Map a Windows group to a Solaris group by adding the Solaris group name to the AD object for the specified Windows group.
- Map a Solaris group to a Windows group by adding the Windows group name to the native LDAP object for the specified Solaris group.

1 **Become superuser, assume an equivalent role, obtain the** `solaris.admin.idmap.rules` **RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine whether to augment a group object in AD or in the native LDAP service.**

- **To augment the Windows group object in AD, type:**

  ```
  # idmap set-namemap wingroup:group-name@domain-name unixgroup:group-name
  ```

For example, the following command maps Windows group `salesgrp@example.com` to Solaris group `sales` by adding the Solaris name to the AD object for `salesgrp@example.com`:

```
# idmap set-namemap wingroup:salesgrp@example.com unixgroup:sales
```

- **To augment the Solaris group object in native LDAP, type:**

  `# idmap set-namemap unixgroup:`*group-name* `wingroup:`*group-name*`@`*domain-name*

  For example, the following command maps Solaris group `sales` to Windows group `salesgrp@example.com` by adding the Windows name to the native LDAP object for `sales`:

  ```
  # idmap set-namemap unixgroup:sales wingroup:salesgrp@example.com
  ```

▼ How to Remove a Directory-Based Name Mapping From a User Object

1. **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **View the directory-based name mapping information for the specified user.**

   `# idmap get-namemap` *username*

3. **Remove the user name stored in the user object of AD or native LDAP.**

   - **Remove the Solaris name from the AD object for the specified user.**

     `# idmap unset-namemap winuser:`*username*`@`*domain-name*

     For example, the following command removes the Solaris name from the AD object for Windows user `danab@example.com`:

     ```
     # idmap unset-namemap winuser:danab@example.com
     ```

   - **Remove the Windows name from the native LDAP object for the specified user.**

     `# idmap unset-namemap unixuser:`*username*

For example, the following command removes the Windows name from the native LDAP object for Solaris user `dana`:

```
# idmap unset-namemap unixuser:dana
```

▼ How to Remove a Directory-Based Name Mapping From a Group Object

1. **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **View the directory-based name mapping information for the specified group.**

   ```
   # idmap get-namemap group-name
   ```

3. **Remove the group name stored in the group object of AD or native LDAP.**

   - **Remove the Solaris name from the AD object for the specified group.**

     ```
     # idmap unset-namemap wingroup:group-name@domain-name
     ```

     For example, the following command removes the Solaris name from the AD object for Windows group `salesgrp@example.com`:

     ```
     # idmap unset-namemap wingroup:salesgrp@example.com
     ```

   - **Remove the Windows name from the native LDAP object for the specified group.**

     ```
     # idmap unset-namemap unixgroup:group-name
     ```

     For example, the following command removes the Windows name from the native LDAP object for Solaris group `sales`:

     ```
     # idmap unset-namemap unixgroup:sales
     ```

Managing Rule-Based Identity Mapping for Users and Groups (Task Map)

Windows systems and Solaris systems use different identity schemes to determine who is permitted to access systems and system objects. When the Solaris CIFS service is integrated into an existing Windows domain, the Solaris user IDs and group IDs must find equivalent Windows SIDs to use for authorization and file access. The Solaris CIFS service uses identity mapping software to perform these tasks.

By default, no rule-based mappings are configured. In this case, non-ephemeral Solaris UIDs and GIDs are mapped to local SIDs. Local SIDs are composed of the server's SID and an RID that is derived algorithmically from the UID or GID. Similarly, domain user and group SIDs are mapped to ephemerally, dynamically allocated UIDs and GIDs. A system administrator can also create a set of rule-based mappings to map users and groups by name. Such rule-based mapping requires that Windows uses Active Directory and that the specified users and groups must already exist.

The following table points to the tasks that you can use to manage rule-based identity mapping for the Solaris CIFS service in a Windows environment. These tasks use the `idmap(1M)` command to manage identity mapping.

Task	Description	For Instructions
Add a user mapping rule.	Use rules to create identity equivalents for Windows users and Solaris users based on the names in the naming services.	"How to Add a User Mapping Rule" on page 50
Add a group mapping rule.	Use rules to create identity equivalents for Windows groups and Solaris groups based on the names in the naming services.	"How to Add a Group Mapping Rule" on page 52
Import rule-based user mappings from the `usermap.cfg` file.	Use this procedure to add one or more user mappings from a `usermap.cfg` file that specifies rule-based mappings.	"How to Import User Mappings From a Rule-Mapping File" on page 54
List all of the mappings.	Use this procedure to review all mappings or to find particular mappings for users and groups.	"How to Show Mappings" on page 55
Show the mapping for a particular identity.	Use this procedure to view how a particular name or ID is mapped.	"How to Show a Mapping for a Particular Identity" on page 56
Show all the established mappings.	Use this procedure to view the mappings stored in the cache.	"How to Show All Established Mappings" on page 56

Task	Description	For Instructions
Remove a user mapping rule.	Use this procedure to remove a rule-based mapping when a user is no longer part of the naming service in your Windows domain.	"How to Remove a User Mapping Rule" on page 57
Remove a group mapping rule.	Use this procedure to remove a rule-based mapping when a group is no longer part of the naming service in your Windows domain.	"How to Remove a Group Mapping Rule" on page 58

For more information about user and group identities, see "Mapping User and Group Identities" on page 29.

For more information about how to determine your identity mapping strategy, see "Creating Your Identity Mapping Strategy" on page 34.

Note – In a cluster configuration, changes made to user maps and to group maps on one server are immediately propagated to the other server.

▼ How to Add a User Mapping Rule

The `idmap` command enables you to create rule-based mappings between Windows users and Solaris users. By default, the Solaris CIFS service uses ephemeral identity mapping.

Shell special characters, such as the double quote character ("), the asterisk character (*), and the backslash character (\), must be quoted when used as user names and domain names.

1 **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine the user names that you want to map.**

a. **Determine the domain and name of the Windows user that you want to map to a Solaris user.**

The Windows user name must be specified by using one of the following formats:

- `winuser:`*username@domain-name*
- `winuser:'`*domain-name\username*`'`

b. **Determine the name of the Solaris user that you want to map to the Windows user.**

   The Solaris user name must be specified by using the format `unixuser:`*username*.

If *username* is the empty string (`""`), mapping is inhibited. Only directional mappings can have an empty string as their target identity. No mapping is created by the identity mapping service, and the `nobody` ID is used for access control. Note that a user name of `""` should not be used to preclude logins by unmapped Windows users.

If *username* uses the wildcard (`*`), it matches all user names that are not matched by other mappings. Similarly, if *username* is the wildcard Windows name (`*@*`), it matches all user names in all domains that are not matched by other mappings.

3 **Create the user mapping.**

By default, identity mappings are bidirectional, which means that the Windows name is mapped to the Solaris name and the Solaris name is mapped to the Windows name. If you want the mapping to be unidirectional, specify the `-d` option.

If *username* uses the wildcard on both sides of the mapping, the user name is the same for both Windows and Solaris users. For example, if the rule is `'*@example.com' == '*'`, the `jp@example.com` Windows user name would match this rule and map to the `jp` Solaris user name.

Caution – Be careful when creating rule-based mappings that use wildcards for the user names. Windows user names are case insensitive, while Solaris user names are case sensitive. Note that the case of Windows names that appear in `idmap` name rules and in `idmap show` commands is irrelevant.

Solaris environments typically use lowercase characters for user names, but uppercase characters are permitted. Therefore, using a wildcard to map Windows names to Solaris user names might not produce the expected results. Rule-based mapping rules that use the `unixuser:*` target map to the Solaris user name as follows:

- Map the canonical Windows name, which uses the found in the directory entry, to the matching Solaris user name.
- If no such Solaris user name exists, fold the case of the canonical Windows name to lower case and use it as the Solaris CIFS user name.

As a result of this differing treatment of case, user names that appear to be alike might not be recognized as matches. You must create rules to handle such pairings of strings that differ only in case. For example, to map Solaris user `Kerry` to Windows user `kerry@example.com`, you must create the following rule:

```
# idmap add winuser:'*@example.com' unixuser:'*'
# idmap add winuser:kerry@example.com unixuser:Kerry
```

- **Create a bidirectional mapping between a Windows user name and a Solaris user name.**

  ```
  # idmap add winuser:username@domain-name unixuser:username
  ```

- **Create a unidirectional mapping between a Windows user name and a Solaris user name.**

  ```
  # idmap add -d winuser:username@domain-name unixuser:username
  ```

- **Create a unidirectional mapping between a Solaris user name and a Windows user name.**

  ```
  # idmap add -d unixuser:username winuser:username@domain-name
  ```

▼ How to Add a Group Mapping Rule

The `idmap` command enables you to create rule-based mappings between Windows groups and Solaris groups. By default, the Solaris CIFS service uses ephemeral identity mapping.

You can also create diagonal mappings to maps between a Windows group and a Solaris user and between a Solaris group and a Windows user. These mappings are needed when Windows uses a group identity as a file owner or a user identity as a file group.

Shell special characters, such as the double quote character ("), the asterisk character (*), and the backslash character (\), must be quoted when used as group names and domain names.

1 **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine the group names that you want to map.**

a. **Determine the domain and name of the Windows group that you want to map to a Solaris group.**

The Windows group name must be specified by using one of the following formats:

- `wingroup:`*group-name*@*domain-name*
- `wingroup:'`*domain-name*\group-name`'`

b. **Determine the name of the Solaris user or group that you want to map to the Windows group.**

The Solaris group name must be specified by using the format `unixgroup:`*group-name*. The Solaris user name must be specified by using the format `unixuser:`*username*.

If *group-name* is the empty string (""), mapping is inhibited.

If *group-name* uses the wildcard (`*`), it matches all group names that are not matched by other mappings. Similarly, if *group-name* is the wildcard Windows name (`*@*`), it matches all group names in all domains that are not matched by other mappings.

3 Create the group mapping.

By default, identity mappings are bidirectional, which means that the Windows group name is mapped to the Solaris group name, and the Solaris group name is mapped to the Windows group name. If you want the mapping to be unidirectional, specify the `-d` option.

If *group-name* uses the wildcard on both sides of the mapping, the group name is the same for both Windows groups and Solaris groups. For example, if the rule is `"*@example.com"` == `"*"`, the `staff@example.com` Windows group name would match this rule and map to the `staff` Solaris group name.

Caution – Be careful when creating rule-based mappings that use wildcards for the group names. Windows group names are case insensitive, while Solaris group names are case sensitive. Note that the case of Windows names that appear in `idmap` name rules and in `idmap show` commands is irrelevant.

Solaris environments typically use lowercase characters for group names, but uppercase characters are permitted. Therefore, using a wildcard to map Windows names to Solaris group names might not produce the expected results. Rule-based mapping rules that use the `unixgroup:*` target map to the Solaris group name as follows:

- Map the canonical Windows name, which uses the found in the directory entry, to the matching Solaris group name.
- If no such Solaris group name exists, fold the case of the canonical Windows name to lower case and use it as the Solaris CIFS group name.

As a result of this differing treatment of case, group names that appear to be alike might not be recognized as matches. You must create rules to handle such pairings of strings that differ only in case. For example, to map Solaris group `Sales` to Windows group `sales@example.com`, you must create the following rule:

```
# idmap add wingroup:'*@example.com' unixgroup:'*'
# idmap add wingroup:sales@example.com unixgroup:Sales
```

- **Create a bidirectional mapping between a Windows group name and a Solaris group name.**

  ```
  # idmap add wingroup:group-name@domain-name unixgroup:group-name
  ```

- **Create a unidirectional mapping between a Windows group name and a Solaris group name.**

  ```
  # idmap add -d wingroup:group-name@domain-name unixgroup:group-name
  ```

- **Create a unidirectional mapping between a Solaris group name and a Windows group name.**

  ```
  # idmap add -d unixgroup:group-name wingroup:group-name@domain-name
  ```

- **Create a diagonal mapping between a Windows group name and a Solaris user name.**

  # idmap add -d wingroup:*group-name***@***domain-name* **unixuser:***username*

- **Create a diagonal mapping between a Solaris group name and a Windows user name.**

  # idmap add -d unixgroup:*group-name* **winuser:***username***@***domain-name*

▼ How to Import User Mappings From a Rule-Mapping File

The `idmap import` command enables you to import a set of rule-based user mappings that are stored in a file.

The `idmap` supports these file formats:

- The NetApp `usermap.cfg` rule-mapping format is as follows:

  windows-username [*direction*] *unix-username*

  windows-username is a Windows user name in either the *domain-name\username* or *username@domain-name* format.

  unix-username is a Solaris user name.

  direction is one of the following:

  - == means a bidirectional mapping, which is the default.
  - => or <= means a unidirectional mapping.

  The IP qualifier is not supported.

- The Samba `smbusers` rule-mapping format is as follows:

  unixname = *winname1* *winname2* ...

  The mappings are imported as unidirectional mappings from one or more Windows names to a Solaris name.

  The format is based on the "username map" entry of the `smb.conf` man page, which is available on the `samba.org` web site. The use of an asterisk (*) for *winname* is supported. However, the `@group` directive and the chaining of mappings are not supported.

  By default, if no mapping entries are in the `smbusers` file, Samba maps a *winname* to the equivalent *unixname*, if any. The following `idmap` command shows this mapping:

  ```
  idmap add -d winuser:"*@*" unixuser:"*"
  ```

1 **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Import the user mappings from standard input or from a file.**

```
# idmap import [-F] [-f file] format
```

For example, suppose that you have a file called `myusermaps` that uses the `usermap.cfg` format to specify the following user name mappings:

```
# cat myusermaps
dana@example.com == dana
danab@example.com => dana
```

Use one of the following commands to add these mappings to the database:

- `# cat myusermaps | idmap import usermap.cfg`
- `# idmap import -f myusermaps usermap.cfg`

▼ How to Show Mappings

The `idmap list` command enables you to view all of the rule-based identity mappings that you created for users and groups. You can also find particular mappings for users and groups.

● **List all of the mappings.**

```
$ idmap list
add winuser:terry@example.com unixuser:terrym
add wingroup:members unixgroup:staff
```

- To optionally list only the user mappings, type:

  ```
  $ idmap list | grep user
  add winuser:terry@example.com unixuser:terrym
  ```

- To optionally list only the group mappings, type:

  ```
  $ idmap list | grep group
  add wingroup:members unixgroup:staff
  ```

▼ How to Show a Mapping for a Particular Identity

The `idmap show` command enables you to view the particular name or ID for a name or ID that you specify.

● **Show the equivalent identity for a particular name or ID.**

$ **idmap show [-c] [-v]** *identity* **[***target-type***]**

By default, the `idmap show` command only shows mappings that have already been established.

For example, to view the SID that is mapped to UID 50000, type:

```
$ idmap show uid:50000 sid
S-1-5-21-726303253-4128413635-1168184439
```

To view the Solaris user name for the Windows user name `terry@example.com`, type:

```
$ idmap show terry@example.com
terrym
```

If you specify the `-c` option, `idmap show` forces the evaluation of rule-based mapping configurations or the dynamic allocation of IDs. This command also shows mapping information when an error occurs to help diagnose mapping problems.

The `-v` option includes additional information about how the identity mapping was generated, which can help with troubleshooting. The following example shows that the mapping is ephemeral and was retrieved from the cache:

```
# idmap show -v sid:S-1-5-21-2949573101-2750415176-3223191819-884217
sid:S-1-5-21-2949573101-2750415176-3223191819-884217 -> uid:2175201213
Source: Cache
Method: Ephemeral
```

For name-based mappings, the `idmap show -v` command shows either the mapping rule or the directory distinguished name with the attribute and value that created the mapping.

▼ How to Show All Established Mappings

The `idmap dump` command enables you to view all of the SID-to-UID and SID-to-GID mappings that are stored in the cache.

- **List all of the mappings in the cache.**

  By default, the `idmap dump` command only lists the mappings themselves. The `-v` option includes additional information about how the identity mapping was generated, which can help with troubleshooting.

  ```
  $ idmap dump
  sid:S-1-5-21-2949573101-2750415176-3223191800-2000    ==     uid:50000
  sid:S-1-5-21-2949573101-2750415176-3223191800-2001    ==     uid:50001
  sid:S-1-5-21-2949573101-2750415176-3223191800-2006    ==     uid:50010
  sid:S-1-5-21-2949573101-2750415176-3223191900-3000    ==     uid:2147491840
  sid:S-1-5-21-2949573101-2750415176-3223191900-3001    ==     gid:2147491342
  sid:S-1-5-21-2949573101-2750415176-3223191700-4000    =>     uid:60001
  sid:S-1-5-21-2949573101-2750415176-3223191700-4001    =>     gid:60001
  sid:S-1-5-21-2949573101-2750415176-3223191800-5000    ==     gid:50000
  sid:S-1-5-21-2949573101-2750415176-3223191800-5001    ==     gid:50001
  ```

  - To optionally list only the user mappings, type:

    ```
    $ idmap dump | grep uid
    sid:S-1-5-21-2949573101-2750415176-3223191800-2000    ==     uid:50000
    sid:S-1-5-21-2949573101-2750415176-3223191800-2001    ==     uid:50001
    sid:S-1-5-21-2949573101-2750415176-3223191800-2006    ==     uid:50010
    sid:S-1-5-21-2949573101-2750415176-3223191900-3000    ==     uid:2147491840
    sid:S-1-5-21-2949573101-2750415176-3223191700-4000    =>     uid:60001
    ```

  - To optionally list only the group mappings, type:

    ```
    $ idmap dump | grep gid
    sid:S-1-5-21-2949573101-2750415176-3223191900-3001    ==     gid:2147491342
    sid:S-1-5-21-2949573101-2750415176-3223191700-4001    =>     gid:60001
    sid:S-1-5-21-2949573101-2750415176-3223191800-5000    ==     gid:50000
    sid:S-1-5-21-2949573101-2750415176-3223191800-5001    ==     gid:50001
    ```

▼ How to Remove a User Mapping Rule

The `idmap` command enables you to remove a rule-based mapping that you created.

1 **Become superuser, assume an equivalent role, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Find the user mapping that you want to remove.**

```
# idmap list
```

For example, to find all user mappings that map to the Solaris user `pat`, type:

```
# idmap list | grep pat
```

3 **Remove one or more user mappings.**

- **Remove any rule-based mapping that involves the specified user name,** *username***.**

  ```
  # idmap remove username
  ```

- **Remove rule-based mappings between** *username1* **and** *username2***.**

  ```
  # idmap remove username1 username2
  ```

- **Remove all rule-based mappings.**

  ```
  # idmap remove -a
  ```

▼ How to Remove a Group Mapping Rule

The `idmap` command enables you to remove a rule-based mapping that you created.

1 **Become superuser, assume an equivalent role, obtain the** `solaris.admin.idmap.rules` **RBAC authorization, or use the "Idmap Service Management" RBAC profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Find the group mapping that you want to remove.**

```
# idmap list
```

For example, to find all unidirectional group mappings that map to the Solaris group `staff`, type:

```
# idmap list | grep staff
```

3 **Remove one or more group mappings.**

- **Remove any rule-based mapping that involves the specified group name,** *group-name***.**

  ```
  # idmap remove group-name
  ```

- **Remove rule-based mappings between** *group-name1* **and** *group-name2***.**

  ```
  # idmap remove group-name1 group-name2
  ```

- **Remove all rule-based mappings.**

  ```
  # idmap remove -a
  ```

CHAPTER 3

Solaris CIFS Service Administration (Tasks)

This chapter provides instructions on how to configure the Solaris CIFS service to run as a standalone server (workgroup mode) or in an existing Windows environment (domain mode). This chapter also describes how to manage CIFS shares to be accessed by CIFS clients.

Currently, the Solaris CIFS service runs only in the global zone.

This chapter covers the following topics:

- "Configuring the WINS Service" on page 62
- "Configuring the Solaris CIFS Service Operation Mode (Task Map)" on page 62
- "Managing CIFS Shares (Task Map)" on page 67
- "Managing CIFS Groups (Task Map)" on page 79
- "Disabling the Samba Service" on page 83

For a high-level overview of the Solaris CIFS service configuration process, see "Configuring the Solaris CIFS Service – Process Overview" on page 16.

Note – CIFS is an enhanced version of the SMB protocol, which allows CIFS clients to access files and resources from the CIFS service. The terms SMB and CIFS can be considered interchangeable.

Up-to-date troubleshooting information is available from the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

For information about installing the Solaris CIFS service packages, see Getting Started With the Solaris CIFS Service wiki on the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

Configuring the WINS Service

This section provides information about configuring the Solaris CIFS service as a client to the WINS service. For information about configuring other applicable services, see "Configuring the Solaris CIFS Service – Process Overview" on page 16.

▼ How to Configure WINS

If you are integrating a Solaris CIFS service in an environment that has a WINS server, you can use Windows Internet Naming Service (WINS) for name resolution.

For information about excluding IP addresses from WINS resolution, see Excluding IP Addresses From WINS Name Resolution in the Solaris CIFS Service Troubleshooting wiki.

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Specify the IP address of the primary WINS server.**

   The primary WINS server is the server consulted first for NetBIOS name resolution.

   ```
   # sharectl set -p wins_server_1=IP-address smb
   ```

3. **(Optional) Specify the IP address of the secondary WINS server.**

   If the primary WINS server does not respond, the system consults the secondary WINS server to perform NetBIOS name resolution.

   ```
   # sharectl set -p wins_server_2=IP-address smb
   ```

Configuring the Solaris CIFS Service Operation Mode (Task Map)

The following table points to the tasks that you can use to configure the operation mode of the Solaris CIFS server.

Task	Description	For Instructions
Configure the Solaris CIFS service in domain mode.	Use the `smbadm join -u` *username* *domain-name* command to join the domain.	"How to Configure the Solaris CIFS Service in Domain Mode" on page 63
Configure the Solaris CIFS service in workgroup mode.	Use the `smbadm join -w` *workgroup-name* command to join the workgroup.	"How to Configure the Solaris CIFS Service in Workgroup Mode" on page 65

▼ How to Configure the Solaris CIFS Service in Domain Mode

After successfully joining an AD domain, you can enable the Solaris CIFS service to publish CIFS shares in the AD directory. To do so, create or update CIFS shares and specify the share container for each share that you want to publish. To create CIFS shares, see "How to Create a CIFS Share (`zfs`)" on page 69 and "How to Create a CIFS Share (`sharemgr`)" on page 74.

Before You Begin If the Samba service is running on the Solaris system, you must disable it. See "How to Disable the Samba Service" on page 83.

If you change from workgroup mode to domain mode, or from domain mode to workgroup mode, you must restart the Solaris CIFS service. To restart the service, run the `svcadm restart smb/server` command.

The *Active Directory (AD)* service is a Windows 2000 namespace that is integrated with the Domain Name Service (DNS). AD runs only on domain controllers. In addition to storing and making data available, AD protects network objects from unauthorized access and replicates objects across a network so that data is not lost if one domain controller fails.

For the Solaris CIFS service to integrate seamlessly into a Windows AD environment, the following must exist on the network:

- A Windows AD domain controller
- An optional Active Directory DNS server that permits dynamic updates to use the dynamic DNS (DDNS) capability

The AD and DDNS clients rely on the Kerberos protocol to acquire the Kerberos ticket-granting ticket (TGT) for the specified AD domain. The system must be configured to use DNS for host lookup.

In order to participate in an AD domain, the system must be configured to use DNS for host lookup. Ensure that the `/etc/nsswitch.conf` and `/etc/resolv.conf` files are configured correctly for the appropriate AD domain.

In the /etc/krb5/krb5.conf file, specify the fully qualified AD domain name, in uppercase characters, as the default realm. Also, specify the fully qualified host name of the domain controller as the value for the kdc, admin_server, and kpasswd_server parameters.

The following example /etc/krb5/krb5.conf file is for an AD domain called EXAMPLE.COM, and the AD domain controller system is called dc.example.com. The fully qualified names are used for the domain and the domain controller.

```
[libdefaults]
   default_realm = EXAMPLE.COM

[realms]
   EXAMPLE.COM = {
       kdc = dc.example.com
       admin_server = dc.example.com
       kpasswd_server = dc.example.com
       kpasswd_protocol = SET_CHANGE
   }

[domain_realm]
   .example.com = EXAMPLE.COM
```

For descriptions of the sections and parameters used in this sample file, see the krb5.conf(4) man page and "Configuring Kerberos Clients (Task Map)" in *System Administration Guide: Security Services*.

1 **Become superuser, assume an equivalent role, obtain the** solaris.smf.value.smb **and** solaris.smf.manage.smb **RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Enable the Solaris CIFS service.**

```
# svcadm enable -r smb/server
```

When you specify the -r option, all services on which smb/server depends are started if they are not already running.

3 **To successfully complete the join process, ensure that the system clock on the Solaris system is within five minutes of the system clock of the domain controller (DC).**

You can accomplish this task in one of these ways:

- **Manually adjust the system clock on either the Solaris system or the DC to match the other.**

- **Configure both the Solaris system and the DC to use the same time source (NTP server).**
- **Synchronize the system clock on the Solaris system with the system clock of the DC by running the following command on the Solaris system:**

  ```
  # ntpdate DC-hostname
  ```

  For example, to synchronize with the DC called `dc.westsales.example.com`, type:

  ```
  # ntpdate dc.westsales.example.com
  ```

4 **Join the Windows domain.**

```
# smbadm join -u username domain-name
```

where *username* is the domain administrator or a user with Domain Administrator privileges, and *domain-name* is a fully qualified NetBIOS or DNS domain name.

Example 3–1 Configuring the Solaris CIFS Service in Domain Mode

This example shows the steps taken to configure the Solaris CIFS service in domain mode. User `dana` has Domain Administrator privileges. The name of the domain being joined is `westsales.example.com`.

```
# svcadm enable -r smb/server
# smbadm join -u dana westsales.example.com
Enter domain password:
Joining 'westsales.example.com' ... this may take a minute ...
Successfully joined domain 'westsales.example.com'
```

▼ How to Configure the Solaris CIFS Service in Workgroup Mode

After you join a workgroup, you can access CIFS shares. To create CIFS shares, see "How to Create a CIFS Share (`zfs`)" on page 69 and "How to Create a CIFS Share (`sharemgr`)" on page 74.

If you change from workgroup mode to domain mode, or from domain mode to workgroup mode, you must restart the Solaris CIFS service. To restart the service, run the `svcadm restart smb/server` command.

Before You Begin If the Samba service is running on the Solaris system, you must disable it. See "How to Disable the Samba Service" on page 83.

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Enable the Solaris CIFS service.**

```
# svcadm enable -r smb/server
```

This command enables the Solaris CIFS service and any service on which it depends, such as the `idmap` service.

3 **(Optional) Change the Solaris CIFS service to operate in a different workgroup.**

By default, the Solaris CIFS service operates in a workgroup called `WORKGROUP`.

```
# smbadm join -w workgroup-name
```

4 **Edit the `/etc/pam.conf` file to support creation of an encrypted version of the user's password for CIFS.**

Add the following line to the end of the file:

```
other    password required    pam_smb_passwd.so.1    nowarn
```

See the `pam_smb_passwd(5)` man page.

5 **Specify the password for existing local users.**

The Solaris CIFS service cannot use the Solaris encrypted version of the local user's password for authentication. Therefore, you must generate an encrypted version of the local user's password for the Solaris CIFS service to use. When the SMB PAM module is installed, the `passwd` command generates such an encrypted version of the password.

```
# passwd username
```

Example 3–2 Configuring the Solaris CIFS Service in Workgroup Mode

This example shows how to configure the Solaris CIFS service in workgroup mode. The name of the workgroup being joined is `myworkgroup`.

```
# svcadm enable -r smb/server
# smbadm join -w myworkgroup
```

Then, use the `sharesmb` property to configure CIFS sharing for an existing ZFS dataset called `ztank/myfs`.

```
# zfs set sharesmb=on ztank/myfs
```

Finally, install the PAM module and generate the password for user `cal`.

```
# passwd cal
```

Now, you are ready to have CIFS clients access the CIFS shares on your Solaris CIFS service.

Managing CIFS Shares (Task Map)

You can add, view, and update CIFS shares. A directory must exist before it can be shared. For more information about CIFS shares, see "CIFS Shares" on page 22.

The following table points to the tasks that you can use to manage CIFS shares.

Task	Description	For Instructions
Configure cross-protocol locking.	Use the `mount` or the `zfs create` command to configure cross-protocol locking. These commands enable this locking by setting the `nbmand` option.	"How to Configure Cross-Protocol Locking" on page 68
Create a CIFS share by using the ZFS file system `sharesmb` property.	Use this procedure if you want to make a file or directory available to clients. You might use this procedure if you are familiar with the ZFS file system `sharenfs` property.	"How to Create a CIFS Share (`zfs`)" on page 69
Create a CIFS share by using the `sharemgr` command.	Use this procedure if you want to make a file or directory available to clients. If you specify an AD container, `sharemgr` will attempt to publish those shares in AD.	"How to Create a CIFS Share (`sharemgr`)" on page 74
Modify the properties of a CIFS share by using the `sharemgr` command.	Use this procedure to change share property values.	"How to Modify CIFS Share Properties (`sharemgr`)" on page 75
Remove a CIFS share by using the `sharemgr` command.	When you remove a share, it can no longer be accessed by a system. If you are connected to the share when it is removed, the share is not removed until there are no more connections to that share. At that time, the share is removed.	"How to Remove a CIFS Share (`sharemgr`)" on page 76

Task	Description	For Instructions
Create an autohome share rule.	Specify custom share rules for autohome shares.	"How to Create a Specific Autohome Share Rule" on page 76
Restrict host access to a share by using the ZFS file system `sharesmb` property.	Use this procedure if you want to restrict access to a client host in one of the following ways: read-write access, read-only access, or no access. You might use this procedure if you are familiar with the ZFS file system `sharenfs` property.	"How to Restrict Client Host Access to a CIFS Share (`zfs`)" on page 77
Restrict host access to a share group by using the `sharemgr` command.	Use this procedure if you want to restrict access to a client host in one of the following ways: read-write access, read-only access, or no access.	"How to Restrict Client Host Access to a CIFS Share (`sharemgr`)" on page 78

▼ How to Configure Cross-Protocol Locking

The CIFS protocol assumes mandatory locking, but UNIX traditionally uses advisory locking. The Solaris OS can be configured to use mandatory locking on a per mount basis by using the non-blocking mandatory locking (`nbmand`) mount option.

When set, the `nbmand` mount option enforces mandatory cross-protocol share reservations and byte-range locking.

When the `nbmand` mount option is not set, the Solaris CIFS service will enforce mandatory share reservations and byte-range locking internally for all CIFS clients. However, without `nbmand` set, there is only limited coordination with NFS and local processes.

1 Become superuser or assume an equivalent role.

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 Set the `nbmand` mount option for an existing file system by doing one of the following:

- **Set the option by using the `mount` command.**

```
# mount -o nbmand=on fsname
```

For example, the following command sets the `nbmand` mount option for the `ztank/myfs` file system:

```
# mount -o nbmand ztank/myfs
```

- **Set the option by using the `zfs create` command.**

  When using the ZFS file system, you can also set the `nbmand` option when the file system is created, so that the file system uses `nbmand` automatically:

  ```
  # zfs create -o nbmand=on fsname
  ```

  The following example combines the `nbmand` option with the mixed-case sensitivity option:

  ```
  # zfs create -o casesensitivity=mixed -o nbmand=on -o mountpoint=mntpt ztank/myfs
  ```

▼ How to Create a CIFS Share (`zfs`)

This procedure describes how to use the ZFS file system `sharesmb` property to create shares on the Solaris CIFS service.

To create an autohome share, you must have defined autohome rules. For more information, see "How to Create a Specific Autohome Share Rule" on page 76.

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Create a ZFS pool and a mixed-case ZFS file system that supports cross-protocol locking.**

```
# zpool create pool-name vdev
# zfs create -o casesensitivity=mixed -o nbmand=on fsname
```

3 **Enable SMB sharing for the ZFS file system.**

```
# zfs set sharesmb=on fsname
```

Note – The resource name for the share is automatically constructed by the `zfs` command when the share is created. The resource name is based on the dataset name, unless you specify a resource name. Any characters that are illegal for resource names are replaced by an underscore character (_).

To specify a resource name for the share, specify a name for the `sharesmb` property, `sharesmb=name=`*resource-name*.

When SMB shares are created on a ZFS file system, the SMB share name appears as an entry in the `.zfs/shares` directory. You can use the `ls` command to show the share-level ACLs on the entries in this directory. You can also use the `chmod` command to modify the share-level ACLs on the entries in this directory. See the `ls(1)` and `chmod(1)` man pages.

4 **Verify how the file system is shared.**

```
# sharemgr show -vp
```

Example 3–3 Using sharesmb to Manage CIFS Shares

The following examples use the sharesmb property to enable SMB sharing for ZFS pools and file systems.

- **Inherited SMB sharing for ZFS file systems in a pool.** The following commands create a pool and enable SMB sharing for that pool. When you create the ZFS file systems in that pool, the file systems inherit the SMB sharing as well as the resource name.

```
# zpool create sandbox c0t3d0
# zfs set sharesmb=on sandbox
# zfs create -o casesensitivity=mixed -o nbmand=on sandbox/fs1
# zfs create -o casesensitivity=mixed -o nbmand=on sandbox/fs2
```

  The sharemgr show -vp command shows how the top level file system has a resource name of sandbox, but the children have the dataset name added to the resource name.

```
# sharemgr show -vp
default nfs=()
mygroup smb=()
zfs nfs=() smb=()
     zfs/sandbox smb=()
           sandbox=/sandbox
           sandbox_fs1=/sandbox/fs1       smb=()
           sandbox_fs2=/sandbox/fs2       smb=()
```

- **SMB sharing for a ZFS file system.** The following commands create a ZFS pool and a mixed-case file system that supports cross-protocol locking:

```
# zpool create sandbox c0t3d0
# zfs create -o casesensitivity=mixed -o nbmand=on sandbox/fs1
```

  Next, you can enable SMB sharing for the sandbox/fs1 file system and for any of its children by setting the sharesmb property to on.

```
# zfs set sharesmb=on sandbox/fs1
```

  Because CIFS shares must have a resource name, the ZFS file system constructs a resource name from the dataset name. Any characters in the dataset name that are illegal in resource names are replaced by the underscore character (_). In this example, the resource name sandbox_fs1 is based on the dataset name sandbox/fs1.

  You can use the sharemgr show -vp command to verify how the file system is shared.

```
# sharemgr show -vp
default nfs=()
mygroup smb=()
zfs nfs=() smb=()
```

```
     zfs/sandbox/fs1 smb=()
           sandbox_fs1=/sandbox/fs1
```

The following commands create another file system in the `sandbox` pool called `fs2` and associate that file system with the `myshare` resource name:

```
# zfs create -o casesensitivity=mixed -o nbmand=on sandbox/fs2
# zfs set sharesmb=name=myshare sandbox/fs2
```

Use the `sharemgr show -vp` command to verify how the file systems are shared.

```
# sharemgr show -vp
default nfs=()
mygroup smb=()
zfs nfs=() smb=()
     zfs/sandbox/fs1 smb=()
           sandbox_fs1=/sandbox/fs1
     zfs/sandbox/fs2 smb=()
           myshare=/sandbox/fs2
```

The following command creates a sub file system of `sandbox/fs2` called `sandbox/fs2/fs2_sub1`:

```
# zfs create -o casesensitivity=mixed -o nbmand=on sandbox/fs2/fs2_sub1
```

This new file system inherits part of its resource name from its parent and also inherits sharing over SMB, if enabled. Because the resource name for `sandbox/fs2` is `myshare`, the resource name for `sandbox/fs2/fs2_sub1` is `myshare_fs2_sub1`.

```
# sharemgr show -vp
default nfs=()
mygroup smb=()
zfs nfs=() smb=()
     zfs/sandbox/fs1 smb=()
           sandbox_fs1=/sandbox/fs1
     zfs/sandbox/fs2 smb=()
           myshare=/sandbox/fs2
           myshare_fs2_sub1=/sandbox/fs2/fs2_sub1
```

If you disable SMB sharing for `sandbox/fs2`, that file system and its children are affected.

```
# zfs set sharesmb=off sandbox/fs2
# sharemgr show -vp
default nfs=()
mygroup smb=()
zfs nfs=() smb=()
     zfs/sandbox/fs1 smb=()
```

```
            sandbox_fs1=/sandbox/fs1
```

The sharemgr show -vp output shows that the sandbox/fs2 file system and its children are no longer shared over SMB.

Example 3–4 Using ls and chmod to Manage CIFS Share-Level ACLs

The following example shows how to view the share-level ACLs on CIFS shares in the .zfs/shares directory. This example also shows how to use the chmod command to modify the ACLs on these shares. Finally, the example shows how to verify that the ACL has been correctly updated by using the ls command. For more information about using the chmod command to modify ACLs, see the chmod(1) man page.

The ACLs are stored on resources located in the .zfs/shares subdirectory in the root of the shared file system. In this example, the shared file system is /zpool/cosmos and one resource, pluto, is stored in the .zfs/shares directory for this file system.

After changing to the /zpool/cosmos/.zfs/shares directory, you can use the ls -lv command to view the ACL information on the resources in that directory.

```
# cd /zpool/cosmos/.zfs/shares
# ls -lv
total 2
----------+  1 root     root           0 Feb  8 18:35 pluto
     0:everyone@:read_data/write_data/append_data/read_xattr/write_xattr
         /execute/delete_child/read_attributes/write_attributes/delete
         /read_acl/write_acl/write_owner/synchronize:allow
```

The ls -lv output shows that the pluto resource is owned by the root user and the root group. The everyone ACL entry covers all other users who are not the root user or part of the root group. The everyone ACL entry shows that everyone has all access privileges, which is the default.

Next, use the chmod command to add a user, terry, who only has read access to the pluto resource. After running the chmod command, the ls -lv command shows you the new ACL entry for user terry. Note that the ACL entry for everyone is unchanged.

```
# chmod A+user:terry:read_data/read_xattr/read_attributes/read_acl:allow pluto
# ls -lv
total 2
-rwxrwxrwx+  1 root     root           0 Feb  8 18:35 pluto
     0:user:terry:read_data/read_xattr/read_attributes/read_acl:allow
     1:everyone@:read_data/write_data/append_data/read_xattr/write_xattr
         /execute/delete_child/read_attributes/write_attributes/delete
         /read_acl/write_acl/write_owner/synchronize:allow
```

Use the `chmod` command to modify the ACL entry for user `terry` to permit all access privileges. Now, the `ls -lv` command shows that the ACL entry for user `terry` has been updated to have all access privileges.

```
# chmod A0=user:terry:read_data/write_data/append_data/read_xattr/ \
write_xattr/execute/delete_child/read_attributes/write_attributes/delete/ \
read_acl/write_acl/write_owner/synchronize:allow pluto
# ls -lv
total 2
-rwxrwxrwx+  1 root     root           0 Feb  8 18:35 pluto
     0:user:terry:read_data/write_data/append_data/read_xattr/write_xattr
         /execute/delete_child/read_attributes/write_attributes/delete
         /read_acl/write_acl/write_owner/synchronize:allow
     1:everyone@:read_data/write_data/append_data/read_xattr/write_xattr
         /execute/delete_child/read_attributes/write_attributes/delete
         /read_acl/write_acl/write_owner/synchronize:allow
```

▼ How to Create a CIFS Share (`sharemgr`)

This procedure describes how to create a share definition on the Solaris CIFS service and make the share available to clients.

To create an autohome share, you must have defined autohome rules. For more information, see "How to Create a Specific Autohome Share Rule" on page 76.

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Define a share in the default share group or in another share group.**

   A share name can include any alphanumeric characters, but not the characters listed here:

   `" / \ [ ] : | + ; , ? * =`

   Create a share group and add a share to that group.

   ```
   # sharemgr create -P smb group-name
   # sharemgr add-share -r resource-name -s share-path group-name
   ```

3. **If AD is enabled, specify the AD container where the share will be published.**

Note – The container must already exist for the share to be published in that container. The system does not create container objects in the AD *tree*.

```
# sharemgr set [-hnv] -P smb [-S option-set] [-p property=value ... \
[-s share-path] group-name
```

▼ How to Modify CIFS Share Properties (`sharemgr`)

Use this procedure to change properties on a share.

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Modify the CIFS share properties.**

   - **Modify properties for a single share.**

     ```
     # sharemgr set-share [-r resource-name] [-d "description"] -s share-path group-name
     ```

     For example, the following command changes the description for the `/files/acme.sales.logs` share in the `nw-sales` group:

     ```
     # sharemgr set-share -d "Sales logs for Acme" -s /files/acme.sales.logs nw-sales
     ```

   - **Modify properties for a share group.**

     ```
     # sharemgr set [-hnv] -P smb [-S option-set] [-p property=value] ... \
     [-s share-path] group-name
     ```

     For example, in domain mode you can configure shares to be published in an AD container. The following command specifies that shares in the `nw-sales` share group will be published in the default container. Note that `filesvr` is the machine account of the system that is running the Solaris CIFS service.

     ```
     # sharemgr set -P smb -p ad-container=cn=filesvr,cn=Computers nw-sales
     ```

     If you want to publish shares to a non-default container, you must modify the ACLs of that container to give the Solaris CIFS service permission to publish and unpublish shares.

▼ How to Remove a CIFS Share (`sharemgr`)

This procedure describes how to remove a CIFS share. When you remove a CIFS share, the definition of the share is removed from the server. You can re-create such a share with the `sharemgr add-share` command.

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Remove a CIFS share.**

   ```
   # sharemgr remove-share -s share-path group-name
   ```

   For example, to remove share `/sales/share1` from share group `mysharegroup`, type:

   ```
   # sharemgr remove-share -s /sales/share1 mysharegroup
   ```

▼ How to Create a Specific Autohome Share Rule

The autohome share feature eliminates the administrative task of defining and maintaining home directory shares for each user that accesses the system through the SMB protocol. The system creates autohome shares when a user logs in, and removes them when the user logs out. This procedure describes how to configure autohome shares by adding rules to a configuration file.

For information about the `smbautohome` format, see "Autohome Entries" on page 24 and the `smbautohome(4)` man page.

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Edit the `/etc/smbautohome` file.**

An autohome entry must be on a single line in the following format:

key *location* [*container*]

a. **Specify the user name in the key field.**

Usually this field is a user name, but it can also be one of the following:

- `+nsswitch` – Uses the naming service to match users to home directories if no rule matches.
- **Asterisk (*)** – Matches a user name to a home directory that uses the same name.

b. **Specify the location of the user's home directory in the location field.**

Specify the absolute path excluding the user name, or use one of the following substitution characters:

- **Question mark (?)** – Substitutes for the first character of the user name.
- **Ampersand (&)** – Substitutes for a complete user name.

For example, the following rule maps to `/home/a/amy`:

```
amy            /home/?/&
```

For more information about the path, see "Autohome Shares" on page 23.

How to Restrict Client Host Access to a CIFS Share (`zfs`)

This procedure describes how to use the ZFS file system `sharesmb` property to restrict access to a share based on a client's host address. This feature is known as host-based access control.

A client host is permitted to have *only one* of the following types of access to a share:

- Read-only access
- Read-write access
- No access

For more information about access lists, see the `sharemgr`(1M) man page.

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine the kind of access you want to grant for each client host.**

3 **Restrict access by particular hosts to a dataset.**

```
# zfs set sharesmb=ro=hostname[:hostname] dataset
# zfs set sharesmb=rw=hostname[:hostname] dataset
# zfs set sharesmb=none=hostname[:hostname] dataset
```

hostname can be a host name, a netgroup, or an IP address. *dataset* is the name of the dataset.

You can specify the host access policy by combining the access settings in a single command. For example, the following command specifies how particular hosts can access `files/acme.sales.logs`. `mercury` and `venus` have read-write access, `mars` has read-only access, and `neptune` has no access.

```
# zfs set sharesmb=rw=mercury:venus,ro=mars,none=neptune files/acme.sales.logs
```

Note – Ensure that your existing dataset property values are not lost when changing the `sharesmb` property for that dataset. If you previously set `sharesmb` property values, specify them all again along with the new value on the `zfs set` command line. If the existing property values are not specified again, the values are lost or reset to default values, if appropriate.

▼ How to Restrict Client Host Access to a CIFS Share (`sharemgr`)

This procedure describes how to use the `sharemgr` command to restrict access to a share group based on a client's host address. This feature is known as host-based access control.

A client host is permitted to have *only one* of the following types of access to a share:

- Read-only access
- Read-write access
- No access

For more information about access lists, see the `sharemgr`(1M) man page.

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine the kind of access you want to grant for each client host.**

3 **Restrict access by particular hosts to a share group.**

```
# sharemgr set -P smb -p ro=hostname[:hostname] group-name
# sharemgr set -P smb -p rw=hostname[:hostname] group-name
# sharemgr set -P smb -p none=hostname[:hostname] group-name
```

hostname can be a host name, a netgroup, or an IP address. *group-name* is the share group.

You can specify the host access policy by combining the access settings in a single command. For example, the following command specifies how particular hosts can access the `nw-sales` share group. `mercury` and `venus` have read-write access, `mars` has read-only access, and `neptune` has no access.

```
# sharemgr set -P smb -p rw=mercury:venus -p ro=mars -p none=neptune nw-sales
```

Managing CIFS Groups (Task Map)

This section describes how to manage CIFS groups and privileges for the Solaris CIFS service.

Note – CIFS groups apply only to users that are connected through CIFS.

For information about CIFS groups and local users, see "Local CIFS Groups" on page 26.

The following table points to the tasks that you can use to manage CIFS groups through the Solaris CIFS service.

Task	Description	For Instructions
Create a CIFS group.	Create a CIFS group to manage users.	"How to Create a CIFS Group" on page 80
Add a member to a CIFS group.	Add a member to a CIFS group by using the `smbadm` command.	"How to Add a Member to a CIFS Group" on page 81
Remove a member from a CIFS group.	Remove a member from a CIFS group by using the `smbadm` command.	"How to Remove a Member From a CIFS Group" on page 82

Task	Description	For Instructions
Modify CIFS group properties.	A CIFS group can grant the following privileges: ■ `backup`. Permit group members to back up file system objects. ■ `restore`. Permit group members to restore file system objects. ■ `take-ownership`. Permit group members to take ownership of file system objects. You can specify a description of the CIFS group if you modify the value of the `description` property.	"How to Modify CIFS Group Properties" on page 82

You use the `smbadm`(1M) command to manage CIFS groups on the system that runs the Solaris CIFS service.

▼ How to Create a CIFS Group

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Choose the name of the group to create.**

You might choose a name that reflects a common set of tasks that the group can perform or the organization to which the group members belong.

3 **Create the CIFS group.**

```
# smbadm create [-d description] group-name
```

The `-d` option is used to specify a textual description of the CIFS group.

For example, to create a group called `wsales`, type:

```
# smbadm create -d "Sales Force for the Western Region" wsales
```

In order to provide proper identity mapping between CIFS groups and Solaris groups, a CIFS group must have a corresponding Solaris group. This requirement has two consequences. First, the group name must conform to the intersection of the Windows and Solaris group name rules. Thus, a CIFS group name can be up to eight (8) characters long and contain only lowercase characters and numbers. Second, a Solaris group has to be created before a CIFS group can be created. The Solaris group is created by using the `groupadd` command. See the `groupadd`(1M) man page.

▼ How to Add a Member to a CIFS Group

1 **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Add a user to the CIFS group.**

```
# smbadm add-member -m member-name [[-m member-name] ...] group-name
```

member-name can be specified as [*domain-name*`\`]*username* or [*domain-name*`/`]*username*. The domain name is the domain in which the user can be authenticated. By default, the domain name is the name of the domain that you joined.

The backslash character (`\`) is a shell special character and must be quoted. For instance, escape the backslash character with another backslash character: *domain*`\\`*username*. For more information about handling shell special characters, see the man page for your shell.

For example, to add user `terry` of the `sales` domain to the `wsales` group, type:

```
# smbadm add-member -m sales\\terry wsales
```

To add a local user to a CIFS group, specify the Solaris host name rather than the domain name. For example, to add local user `terry` of the `solarsystem` host to the `wsales` group, type:

```
# smbadm add-member -m solarsystem\\terry wsales
```

▼ How to Remove a Member From a CIFS Group

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Remove a user from the CIFS group.**

   ```
   # smbadm remove-member -m member-name [[-m member-name] ...] group-name
   ```

   member-name can be specified as [*domain-name*\]*username* or [*domain-name*/]*username*. The domain name is the domain in which the user can be authenticated. By default, the domain name is the name of the domain that you joined.

   The backslash character (\) is a shell special character and must be quoted. For instance, escape the backslash character with another backslash character: *domain*\\username. For more information about handling shell special characters, see the man page for your shell.

   For example, to remove user `terry` of the `sales` domain from the `wsales` group, type:

   ```
   # smbadm remove-member -m sales\\terry wsales
   ```

   To remove a local user from a CIFS group, specify the Solaris host name rather than the domain name. For example, to remove local user `terry` of the `solarsystem` host from the `wsales` group, type:

   ```
   # smbadm remove-member -m solarsystem\\terry wsales
   ```

▼ How to Modify CIFS Group Properties

1. **Become superuser, assume an equivalent role, obtain the `solaris.smf.value.smb` and `solaris.smf.manage.smb` RBAC authorizations, or use the "SMB Management" RBAC profile, which is part of the "File System Management" profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2. **Modify one or more CIFS group properties.**

   ```
   # smbadm set -p property=value [[-p property=value] ...] group-name
   ```

You can specify one or more property-value pairs on the command line. Each property-value pair must be preceded by the `-p` option. Valid values for privileges are `on` or `off`. The value of the `description` property is an arbitrary text string.

For example, to grant the `backup` privilege and to modify the description of the `wsales` group, type:

```
# smbadm set -p backup=on \
-p description="Sales force for the Western region" wsales
```

Disabling the Samba Service

The Samba and CIFS services cannot be used together on a single Solaris system. If you want to run the Solaris CIFS service, you must first ensure that a running Samba service is disabled.

If your Solaris system is running the Samba service, disable it before starting the Solaris CIFS service.

▼ How to Disable the Samba Service

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Verify that the Samba service is running.**

```
# svcs | grep samba
```

For example, the following command shows that the Samba service is running:

```
# svcs | grep samba
legacy_run     Aug_03   lrc:/etc/rc3_d/S90samba
```

3 **Disable the Samba service.**

```
# svcadm disable svc:/network/samba
# svcadm disable svc:/network/wins
```

CHAPTER 4

Solaris CIFS Client Administration (Tasks)

This chapter provides instructions on how to use the Solaris CIFS client to access CIFS shares from a CIFS server in a Windows environment.

This chapter covers the following topics:

- "Managing CIFS Mounts in Your Local Environment (Task Map)" on page 85
- "Managing CIFS Mounts in the Global Environment (Task Map)" on page 93

Note – CIFS is an enhanced version of the SMB protocol, which allows CIFS clients to access files and resources on CIFS servers. The terms SMB and CIFS can be considered interchangeable.

Up-to-date troubleshooting information is available from the OpenSolaris CIFS Server project page (`http://opensolaris.org/os/project/cifs-server/docs`).

Managing CIFS Mounts in Your Local Environment (Task Map)

The following table points to the tasks that a regular user can perform to manage CIFS mounts.

Task	Description	For Instructions
Find the shares that are available on a CIFS server in your domain.	From a particular CIFS server, view the shares that you can mount on a directory that you own.	"How to Find Available CIFS Shares on a Known File Server" on page 86
Mount a CIFS share on a directory that you own.	Use the `mount` command to mount the share on a mount point that you own.	"How to Mount a CIFS Share on a Directory You Own" on page 88

Task	Description	For Instructions
View the list of CIFS shares that are mounted on the system.	View the list of mounted CIFS shares.	"How to View the List of Mounted CIFS Shares" on page 88
Unmount a CIFS share from a directory that you own.	When you no longer need access to a CIFS share, you can unmount it.	"How to Unmount a CIFS Share From a Directory You Own" on page 89
Store a persistent password to be used for authentication.	When you store a persistent password, you can bypass the manual authentication required each time that you want to mount a share from the specified server.	"How to Store a CIFS Persistent Password" on page 89
Use a PAM module to store a persistent password to be used for authentication.	Use this optional functionality only in environments that do not run Active Directory or Kerberos, but which synchronize passwords between Solaris clients and their CIFS/SMB servers.	"How to Configure the PAM Module to Store a CIFS Persistent Password" on page 90
Delete a persistent password.	If you no longer want to store a persistent password, delete it.	"How to Delete a CIFS Persistent Password" on page 92
Customize your environment by using a `$HOME/.nsmbrc` file.	You can customize your Solaris CIFS environment by specifying values for Solaris CIFS client properties.	"How to Customize Your Solaris CIFS Environment" on page 93

▼ How to Find Available CIFS Shares on a Known File Server

1. **Determine the server that you want to query about available shares.**

   If you are not familiar with the CIFS file servers available in your domain, contact your system administrator. You might be able to use Network Neighborhood on Windows systems or the GNOME file browser to browse for available CIFS shares.

2. **List the available CIFS shares on a server.**

   ```
   $ smbutil view [-A | -U user] //[domain;][user[:password]@]server
   ```

   `//[`*domain*`;][`*user*`[:`*password*`]@]`*server* is a resource name. *user* is the user name with which you connect to the CIFS server, *server*. You can optionally specify the domain name and the password of the user that you specified on the command line.

The `-A` option enables you to view shares anonymously, and you are not prompted for a password. The `-U` *user* option indicates the user with which to authenticate on the specified server.

3 **When prompted, enter the password for the user that you specified on the CIFS server.**

If you specified the `-A` option to view shares anonymously, you are not prompted for a password.

If you did not specify a user, enter the password associated with your user name.

4 **View the list of available CIFS shares.**

The `smbutil view` output shows the name of the share, its type, and an optional text description of the share.

Most shares have a type of `disk` because the shares are files and directories. The other share types are as follows:

- `IPC` – Represents an interprocess communication (IPC) device, such as a pipe or a mailslot
- `printer` – Represents a printer queue
- `device` – Represents a communications device

For example, the following command shows how to view the shares on the `solarsystem` server:

```
$ smbutil view //cal@solarsystem
Password:
Share        Type       Comment
-------------------------------
netlogon     disk       Network Logon Service
ipc$         IPC        IPC Service (Samba Server)
tmp          disk       Temporary file space
public       disk       Public Stuff
ethereal     disk
root         disk       Home Directories

6 shares listed from 6 available
```

Note – The Solaris CIFS client does not support `device` shares.

The following command enables you to anonymously view the shares on the `solarsystem` server:

```
$ smbutil view -A //solarsystem
```

▼ How to Mount a CIFS Share on a Directory You Own

Note – If you own the directory on which you want to mount a share, you can perform the mount operation yourself. If you do not own the directory, you must perform the mount operation as the owner of the directory or as superuser.

1 **Verify that the `network/smb/client` service is enabled.**

```
$ svcs network/smb/client
STATE          STIME    FMRI
online         19:24:36 svc:/network/smb/client:default
```

This service is enabled by default, so the usual state for the service is `online`. To enable the service, type the following command:

```
$ svcadm enable network/smb/client
```

2 **Find the share that you want to mount from a server.**

```
$ smbutil view //server
```

3 **Enter your password at the prompt.**

4 **Perform the mount on your directory.**

```
$ mount -F smbfs //[workgroup;][user[:password]@]server/share  mount-point
```

For example, to mount the `/tmp` share from the `solarsystem` server on the `/mnt` mount point, type:

```
$ mount -F smbfs //solarsystem/tmp /mnt
```

▼ How to View the List of Mounted CIFS Shares

This procedure shows how to list all of the CIFS shares that are mounted on your system. The resulting list includes your mounts, other users' mounts, and multiuser mounts created by the system administrator.

● **List all CIFS mounts.**

Use one of the following commands to list the mounted CIFS shares:

- **Use the `mount` command.**

```
$ mount -v | grep ’type smbfs’
//root@solarsystem/tmp on /mnt type smbfs read/write/setuid/devices/dev=5080000
  on Tue Feb 12 11:40:18 2008
```

```
//root@solarsystem/files on /files type smbfs read/write/setuid/devices/dev=4800000
  on Mon Feb 11 22:17:56 2008
```

Note that the `mount` command includes information about the mount options specified at mount time.

- **Use the `df -k -F smbfs` command.**

  ```
  $ df -k -F smbfs
  //root@solarsystem/tmp      1871312   70864 1800448     4%    /mnt
  //root@solarsystem/files    8067749    8017 7979055     1%    /files
  ```

▼ How to Unmount a CIFS Share From a Directory You Own

To successfully unmount a share, you must own the mount point on which the share is mounted.

1 **Determine the mount point of the share that you want to unmount.**

Use one of the following commands to find shares that are mounted from a CIFS server:

- **Use the `mount` command.**

  ```
  $ mount -v | grep 'type smbfs'
  //root@solarsystem/tmp on /mnt type smbfs read/write/setuid/devices/dev=5080000
    on Tue Feb 12 11:40:18 2008
  //root@solarsystem/files on /files type smbfs read/write/setuid/devices/dev=4800000
    on Mon Feb 11 22:17:56 2008
  ```

- **Use the `df -k -F smbfs` command.**

  ```
  $ df -k -F smbfs
  //root@solarsystem/tmp      1871312   70864 1800448     4%    /mnt
  //root@solarsystem/files    8067749    8017 7979055     1%    /files
  ```

2 **Unmount the share by specifying the name of the mount point, `/mnt` or `/files` in the previous step.**

For example:

```
$ umount /mnt
```

▼ How to Store a CIFS Persistent Password

Interactions with a CIFS file server require authentication. For instance, when you view the shares available on a server or you try to mount a share on your system, the transaction is authenticated.

Note – A persistent password is not needed when Kerberos is configured on the client and server and you have a Kerberos ticket-granting ticket (TGT). In such configurations, you can view and mount shares without specifying a password.

You can supply the password each time that you make a connection to the server, or you can store a *persistent password* to be automatically used for these transactions.

Note – You can store a persistent password for each user on the CIFS server that you use to access shares.

The password you store persists until any of the following occur:

- The CIFS client is rebooted.
- The `smbutil logout` command is run for the user.
- The `smbutil logoutall` command is run by superuser.

● **Store the persistent password for the CIFS server.**

```
$ smbutil login user
Password:
```

The following command stores the persistent password for `terry@solarsystem`. Each time Terry performs a transaction with `solarsystem`, the persistent password is used to perform the authentication.

```
$ smbutil login terry@solarsystem
Password:
```

▼ How to Configure the PAM Module to Store a CIFS Persistent Password

When installed, the `pam_smbfs_login.so.1` module enables you to store a persistent password the same as if you had run the `smbutil login` command for `PAM_USER` in the user's or system's default domain.

This optional functionality is meant to be used only in environments that do not run Active Directory or Kerberos, but which synchronize passwords between Solaris clients and their CIFS/SMB servers.

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

1 **Use your login name and password to store a persistent password.**

Add the following line to the `/etc/pam.conf` file after the other `login` entries:

```
login   auth optional           pam_smbfs_login.so.1
```

This action adds a persistent password entry as if you had run the `smbutil login` command.

Note – The PAM module implements a privilege to permit it to run as superuser to store your password.

---

2 **Verify that your persistent password is stored.**

```
$ smbutil login -c user
```

Example 4–1 Configuring the PAM Module to Store a Persistent Password

The following example shows how the domain is chosen. The system default is `WORKGROUP`. The `WORKGROUP` domain is overridden by any default from SMF, and finally by any default from the user's `.nsmbrc` file.

This example shows a default domain in SMF and for user `terry`:

```
# sharectl set -p section=default -p domain=AAA smbfs
# sharectl get smbfs
[default]
domain=AAA
```

A `root` login uses the domain from SMF:

```
# smbutil login -c terry
Keychain entry exists for AAA/terry.
```

A login as `terry` uses the domain from the `~terry/.nsmbrc` file:

```
$ ls /.nsmbrc
/.nsmbrc: No such file or directory

$ cat ~/.nsmbrc
[default]
domain=MYDOMAIN
$ ls -l ~/.nsmbrc
-rw-r--r--   1 terry  staff         26 Feb 13 10:15 /home/terry/.nsmbrc
$ smbutil login terry
Keychain entry exists for MYDOMAIN/terry.
```

If Terry puts a password in `~terry/.nsmbrc`, he must remove read permission. Also, because Terry's home directory is on an NFS server, the PAM module running as `root` cannot access Terry's file, so Terry would see the following and use the SMF domain instead:

```
$ chmod 400 .nsmbrc
$ logout

solarsystem console login: terry
Password:
Can't open /home/terry/.nsmbrc: Permission denied
$ su
Password:
# smbutil login -c terry
Keychain entry exists for AAA/terry.
```

▼ How to Delete a CIFS Persistent Password

Use this procedure to delete persistent passwords that are stored by the `smbutil login` command.

If you want to delete *all* persistent passwords, see "How to Delete All CIFS Persistent Passwords" on page 98.

- **Delete a persistent password for the specified server by doing one of the following:**
  - **To delete the persistent password for a specified user, type:**

    ```
    $ smbutil logout user@server
    ```

    For example, the following command removes the persistent password for `terry@solarsystem`:

    ```
    $ smbutil logout terry@solarsystem
    ```

    After the password is deleted, Terry is prompted for his password each time that he performs a transaction with `solarsystem`.

  - **To delete the password for the user running the `smbutil logout` command, type:**

    ```
    $ smbutil logout server
    ```

    For example, when user `dana` runs the following command, he removes his persistent password for `solarsystem`:

    ```
    $ smbutil logout solarsystem
    ```

    After the password is deleted, Dana is prompted for his password each time that he performs a transaction with `solarsystem`.

▼ How to Customize Your Solaris CIFS Environment

You can customize your Solaris CIFS environment by creating a `.nsmbrc` configuration file in your home directory. For more information about the `.nsmbrc` file format, see the `nsmbrc(4)` man page.

1 **Create a file called `.nsmbrc` file in your home directory.**

2 **Edit the `.nsmbrc` file to specify values for Solaris CIFS client properties.**

This example shows how user `terry` can configure the `example.com` environment by placing this `.nsmbrc` configuration file in his home directory.

The default section describes the default domain, which is called `SALES`, and sets a default user of `MYUSER`. These default settings are inherited by other sections unless property values are overridden.

`FSERVER` is a server section that defines a server called `fserv.example.com`. It is part of the `SALES` domain.

`RSERVER` is a server section that defines a server called `rserv.example.com` that belongs to a new domain called `REMGROUP`.

```
# Configuration file for example.com
# Specify the Windows account name to use everywhere.
[default]
domain=SALES
user=MYUSER

# The ’FSERVER’ is server in our domain.
[FSERVER]
addr=fserv.example.com

# The ’RSERVER’ is a server in another domain.
[RSERVER]
domain=REMGROUP
addr=rserv.example.com
```

Managing CIFS Mounts in the Global Environment (Task Map)

The following table points to the tasks that superuser can perform to manage CIFS mounts.

Task	Description	For Instructions
Mount a share on a public mount point, such as one in the root file system, so that many users can access the share.	Some shares include files and directories that many people on a system might want to access, such as a global set of files or programs. In such cases, instead of each user mounting the share in his own directory, the system administrator can mount the share in a public place so that all users can access the share from the same location.	"How to Mount a Multiuser CIFS Share" on page 94
Customize the global environment by using the `sharectl` command to set Solaris CIFS properties.	User-specified properties override global properties with the exception of security settings.	"How to Customize the Global Solaris CIFS Environment" on page 95
View the global Solaris CIFS property settings by using the `sharectl` command.	If one property is set with different values in each section, all values are shown.	"How to View the Global Solaris CIFS Environment Property Settings" on page 96
Add a CIFS share to an automounter map.	Use this procedure if you want a CIFS share to be automatically mounted at boot time.	"How to Add an Automounter Entry for a CIFS Share" on page 96
Delete all persistent passwords.	Use this procedure if you want to clear all persistent passwords.	"How to Delete All CIFS Persistent Passwords" on page 98

▼ How to Mount a Multiuser CIFS Share

If you want to make a share available to one or more users on a system, you can mount the share on a mount point anywhere on the system. When you mount a share as superuser, you do not need to own the mount point.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Verify that the `network/smb/client` service is enabled.**

```
# svcs network/smb/client
STATE          STIME    FMRI
online         19:24:36 svc:/network/smb/client:default
```

This service is enabled by default, so the usual state for the service is `online`. To enable the service, type the following command:

```
# svcadm enable network/smb/client
```

3 **Find the share that you want to mount from a server.**

```
# smbutil view //server
```

4 **Specify the password at the prompt.**

5 **Determine the mount point that you want to use.**

For example, you decide to mount shares on the `/sales-tools` mount point.

6 **Perform the mount.**

```
# mount -F smbfs //[workgroup;][user[:password]@]server/share  mount-point
```

For example, to mount the `/tmp` share from the `solarsystem` server on the `/sales-tools` mount point, type:

```
# mount -F smbfs //solarsystem/tmp /sales-tools
```

▼ How to Customize the Global Solaris CIFS Environment

You can customize the global Solaris CIFS environment by using the `sharectl`(1M) command. With the exception of the `minauth` property, globally set properties can be overridden by a value set in user's `.nsmbrc` file. The most secure value of the `minauth` property takes precedence over a less secure value set by the user or set in the global environment.

1 **Become superuser, assume an equivalent role, or use the "SMBFS Management" RBAC profile, which is part of the "File System Management" profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Determine which properties you want to set.**

For a description of the properties, see the `nsmbrc`(4) man page.

3 **Set a property value for the global Solaris CIFS environment.**

```
# sharectl set [-h] [-p property=value] ... smbfs
```

For example, to specify a default workgroup name of SALES for the default section, type:

```
# sharectl set -p section=default -p workgroup=SALES smbfs
```

▼ How to View the Global Solaris CIFS Environment Property Settings

You can view the global Solaris CIFS environment property settings by using the `sharectl`(1M) command.

If you set a value for the same property in more than one section, the `sharectl get` output includes the section name, property name, and value.

- **Determine which properties you want to view.**

  For a description of the properties, see the `nsmbrc`(4) man page.

  - **To view the value for a specific property, type:**

    ```
    $ sharectl get [-p property] ... smbfs
    ```

    For example, to view the values for the `timeout` property, type:

    ```
    $ sharectl get -p timeout smbfs
    [SALES] timeout=5
    [default] timeout=10
    ```

  - **To view all of the property settings, type:**

    ```
    $ sharectl get smbfs
    [SALES]
    password=$$178465324253e0c07
    timeout=5

    [default]
    timeout=10
    ```

▼ How to Add an Automounter Entry for a CIFS Share

You can add a CIFS share to an automount map, such as the `/etc/auto_direct` file, so that the share will be automatically mounted when a user accesses the mount point. You cannot add these automount entries to the `/etc/auto_master` file.

To successfully use the automount feature, you must store a persistent password for authentication to mount the share. See "How to Store a CIFS Persistent Password" on page 89.

Caution – When a user mounts a remote CIFS share by using `smbfs`, all accesses through that mount, even by other users, are as the user who established the mount.

For shares that will only be used by the owner, you should restrict access to the share by using the `dirperms` mount option to ensure that only the owner can access the share.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Edit the `/etc/auto_master` file to refer to the automount map.**

For example, to add automount entries to the `/etc/auto_direct` file, add the following line to the `/etc/auto_master` file:

```
/-    auto_direct
```

3 **Edit the automount map to add the mapping.**

The following examples show the changes to the automount map, in this example the `/etc/auto_direct` file, to configure automount maps.

- To configure a private automount (a share that will only be accessed by the owner) of the `//solarsystem/test` share on the `/sam-test` mount point, create the following entry in the `/etc/auto_direct` file:

  ```
  /sam-test -fstype=smbfs,dirperms=0700,uid=sam //solarsystem/test
  ```

  The `dirperms=0700` mount option ensures that only the owner can access the share. The `uid=sam` mount option ensures that the share root and everything in the share is owned by user `sam`.

- To configure a public automount of the `//solarsystem/public` share on the `/PUBLIC` mount point, create the following entry in the `/etc/auto_direct` file:

  ```
  /PUBLIC -fstype=smbfs //solarsystem/public
  ```

  The `dirperms=0555` mount option ensures that everyone has read and execute access to the share.

- To configure a public automount of a share and to specify the password to be used for authentication, create the following entry in the `/etc/auto_direct` file:

  ```
  /PUBLIC -fstype=smbfs //guest:guest@solarsystem/public
  ```

This entry specifies that all access to the `//solarsystem/public` share is done as the user `guest` and uses the specified password, which in this example is `guest`. The `dirperms=0777` mount option ensures that everyone has read, write, and execute access to the share.

- To configure a public automount of a share that can be accessed anonymously, which does not require a password, specify the `noprompt` option:

  ```
  /PUBLIC -noprompt,fstype=smbfs //solarsystem/public
  ```

  The `noprompt` mount option suppresses the prompting for a password when mounting the share. The `dirperms=0555` mount option ensures that everyone has read and execute access to the share.

4 **Run the `automount` command to read the `/etc/auto_master` file.**

```
# automount
```

5 **Access the automounted share.**

The share is automounted when a user accesses the mounted share, such as by using the `ls` or `cd` command.

```
$ ls /PUBLIC
bin docs
```

After the CIFS share is mounted, a user can use regular Solaris commands to access the files. Automounted shares are automatically unmounted after a period of inactivity.

▼ How to Delete All CIFS Persistent Passwords

Use this procedure to delete all of the persistent passwords that are used to authenticate CIFS transactions.

If you only want to delete the persistent passwords for a particular user, see "How to Delete a CIFS Persistent Password" on page 92.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*. To configure a role with the Primary Administrator profile, see Chapter 2, "Working With the Solaris Management Console (Tasks)," in *System Administration Guide: Basic Administration*.

2 **Delete all of the persistent passwords.**

```
# smbutil logoutall
```

After the persistent passwords are deleted, each time a user performs a transaction with a CIFS server, he is prompted for his password.

Glossary

The following terms are used throughout this book.

access control list (ACL)	A list associated with a file that contains information about which users or groups have permission to access or modify the file.
Active Directory (AD)	A Windows naming service that runs on a domain controller to protect network objects from unauthorized access. This service also replicates objects across a network so that data is not lost if one domain controller fails.
autohome share	A transient share of a user's home directory that is created when the user logs in and is removed when the user logs out.
CIFS client	Software that enables a system to access CIFS shares from a CIFS server.
CIFS server	Software that enables a system to make CIFS shares available to CIFS clients.
Common Internet File System (CIFS)	A protocol that follows the client-server model to share files and services over the network, and which is based on the Server Message Block (SMB) protocol.
diagonal mapping	A rule that maps between a Windows group and a Solaris user and between a Solaris group and a Windows user. These mappings are needed when Windows uses a group identity as a file owner, or a user identity as a file group.
directory-based mappings	A way to use name mapping information that is stored in user or group objects in the Active Directory (AD), in the native LDAP directory service, or both to map users and groups.
Domain Name System (DNS)	A service that provides the naming policy and mechanisms for mapping domain and machine names to addresses outside of the enterprise, such as those on the Internet. DNS is the network information service used by the Internet.
Dynamic DNS (DDNS)	A service that is provided with AD that enables a client to dynamically update its entries in the DNS database.
ephemeral ID	A dynamic UID or GID mapping for an SID that is not already mapped by name.
forest	A forest can have one or more trees that do not form a contiguous namespace.
forest-and-tree model	A logical structure that enables you to interconnect two or more Windows domains by bringing them into bidirectional, chained trust relationships. See also *tree* and *forest*.

	Each tree in this model has a unique name, while a forest does not need to be named. The trees in a forest form a hierarchy for the purposes of the trust relationships. In this model, a single tree can constitute a forest. Each tree within a forest can be independent of the others. You might use this model to run multiple environments under separate DNS namespaces.
group identifier (GID)	An unsigned 32-bit identifier that is associated with a Solaris group.
identity mapping	A process that enables Windows clients to transparently access CIFS shares and remote services from the Solaris CIFS server.
Lightweight Data Access Protocol (LDAP)	A standard, extensible directory access protocol that enables clients and servers that use LDAP naming services to communicate with each other.
mount point	A directory to which you mount a file system or a share that exists on a remote system.
name-based mappings	A way to associate Windows users and groups with equivalent Solaris users and groups by name rather than by identifier. A name-based mapping can consist of directory-based mappings and rule-based mappings.
NetBIOS name	The name of a host or workgroup used by NetBIOS.
NetBIOS scope	A valid domain name as defined by DNS. You use a NetBIOS scope identifier to identify logical NetBIOS networks that are on the same physical network. When you specify a NetBIOS scope identifier, the server will only be able to communicate with other systems that have the same scope defined. The value is a text string that represents a domain name and is limited to 16 characters. By default, no value is set. You might specify a NetBIOS scope if you want to divide a large Windows workgroup into smaller groups. If you use a scope, the scope ID must follow NetBIOS name conventions or domain name conventions. The ID is limited to 16 characters. Most environments do not require the use of the NetBIOS scope feature. If you must use this feature, ensure that you track the scope identifier assigned to each node.
Network Information Service (NIS) database	A distributed database that contains key information about the systems and the users on the network. The NIS database is stored on the master server and all the replica or slave servers.
Network Time Protocol (NTP)	A protocol that enables a client to automatically synchronize its system clock with a time server. The clock is synchronized each time the client is booted and any time it contacts the time server.
persistent password	A stored password that enables a Solaris CIFS client to mount CIFS shares without having to authenticate each mount action. This password remains in storage until removed by the `smbutil logout` or `smbutil logoutall` command.
relative identifier (RID)	A 32-bit identifier similar to a Solaris user identifier (UID) or group identifier (GID) that identifies a user, group, system, or domain.

rule-based mappings	A way to use rules to associate Windows users and groups with equivalent Solaris users and groups by name rather than by identifier.
Samba	An open source service that enables UNIX servers to provide CIFS/SMB file-sharing and printing services to CIFS clients.
Security Accounts Manager (SAM) database	A database in which Windows users and groups are defined. The SAM database is managed on a Windows domain controller.
security identifier (SID)	A variable length structure that uniquely identifies a user or group both within the local domain and across all possible Windows domains.
Server Message Block (SMB)	A protocol that enables clients to access files and to request services of a server on the network.
share	A local resource on a server that is accessible to clients on the network. On a Solaris CIFS server, a share is typically a directory. Each share is identified by a name on the network. To clients on the network, the share does not expose the local directory path directly above the root of the share. Most shares have a type of `disk` because the shares are directories. A share of type `pipe` represents a device, such as an IPC share or a printer.
tree	A named collection of domains that share the same network configuration, schema, and global catalog.
user identifier (UID)	An unsigned 32-bit identifier that is associated with a Solaris user.
Windows domain	A centrally administered group of computers and accounts that share a common security and administration policy and database. Computer, user, and group accounts are centrally managed by using servers known as domain controllers. In order to participate in a Windows domain, a computer must join the domain and become a domain member.
Windows domain controller	A Windows system that is used to provide authentication services for its Windows domain.
Windows Internet Naming Service (WINS)	A service that resolves NetBIOS names to IP addresses.
Windows workgroup	A group of standalone computers that are independently administered. Each computer has independent, local user and group accounts, and security and policy database. In a Windows workgroup, computers cooperate through the use of a common workgroup name but this is a peer-to-peer model with no formal membership mechanism.

Index