Oracle® Solaris Administration: SMB and Windows Interoperability

Part No: 821–1449–10
November 2011

111206@25097

Contents

Preface

The *Oracle Solaris Administration: SMB and Windows Interoperability* describes the Oracle Solaris Server Message Block (SMB) server. This book is intended for system administrators and end users. Both Oracle Solaris operating system (Oracle Solaris OS) and Windows system administrators can use this information to configure and integrate the SMB server into a Windows environment. In addition, system administrators can configure the identity mapping service. Finally, the chapter about the SMB client is primarily intended for Oracle Solaris users who would like to mount SMB shares. The SMB client chapter also includes tasks to be performed by a system administrator.

Access to Oracle Support

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

Typographic Conventions

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

TABLE P–1 Typographic Conventions

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

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

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

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

TABLE P–2 Shell Prompts

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

CHAPTER 1

Windows Interoperability (Overview)

This administration guide provides the information needed to integrate an Oracle Solaris Server Message Block (SMB) server into an existing Windows environment and also describes the SMB client, which enables you to mount SMB shares on Oracle Solaris systems.

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

This chapter covers the following topics:

- "The SMB Server" on page 10
- "Configuring the SMB Server – Process Overview" on page 14
- "Utilities and Files Associated With the SMB Server and Client" on page 15
- "Authentication, Directory, Naming, and Time Services" on page 19
- "SMB Shares" on page 19
- "Local SMB Groups" on page 24
- "Client-Side Caching for Offline Files" on page 25
- "Share Execution Properties" on page 28
- "Support for the Distributed File System" on page 29
- "Support for SMB Printing" on page 29

Note – The Oracle Solaris OS provides a *Server Message Block (SMB)* protocol server and client implementation that includes support for numerous SMB dialects including NT LM 0.12 and Common Internet File System (CIFS). The terms CIFS and SMB can be considered interchangeable.

Up-to-date troubleshooting information is available on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

For information about installing the SMB server packages, see Getting Started With the Oracle Solaris SMB Service wiki on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

The SMB Server

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

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

An SMB server can operate in either workgroup mode or in domain mode. In workgroup mode, the SMB server 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 SMB server uses pass-through authentication, in which user authentication is delegated to a domain controller.

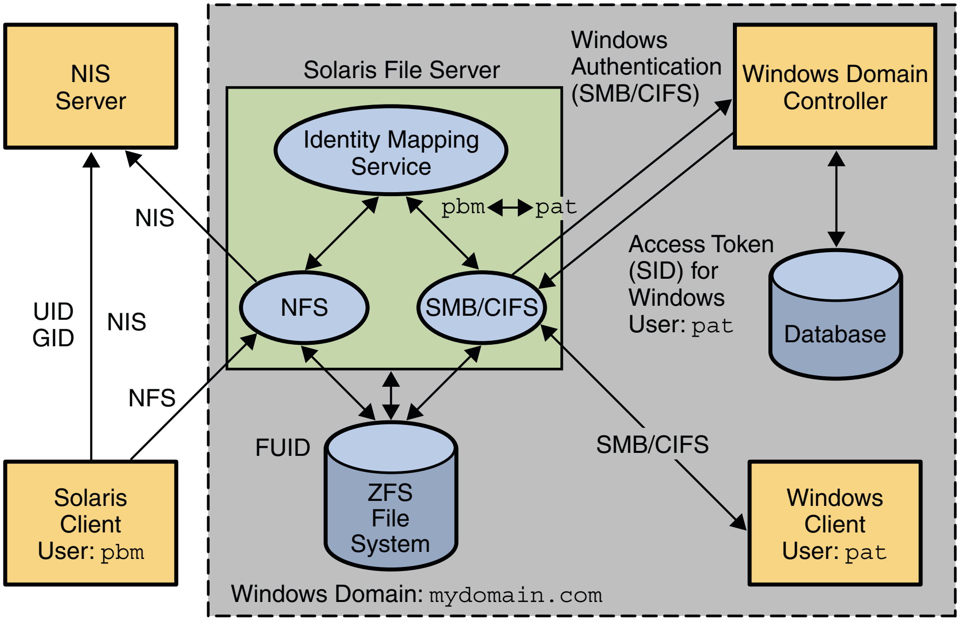
When a user is successfully authenticated, the SMB server generates an access token using the security identifiers (SIDs) that represent the user's identity and the groups of which the user is a member. When the user requests access to files or resources from the server, the access token is used to determine access to files by cross-checking the token with the access control list (ACL) or permissions on files and resources. Oracle 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 Oracle 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 SMB server, the user's SMB 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 Oracle 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 SMB server, the SIDs are used to generate the access control entries. Oracle Solaris utilities, such as `ls` and `chmod`, support ACL management.

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

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

FIGURE 1–1 SMB Environment



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

- “SMB Server” on page 12
- “SMB Client” on page 13
- “Identity Mapping Service” on page 13

SMB Server

Note – *Samba* and SMB servers cannot be used simultaneously on a single Oracle Solaris system. The Samba server must be disabled in order to run the SMB server. For more information, see "How to Disable the Samba Service" on page 60.

For a high-level overview of configuring the SMB server, see "Configuring the SMB Server – Process Overview" on page 14. For information about configuring the server, see Chapter 3, "SMB Server Administration (Tasks)." For more information about the SMB server, see the `smbadm`(1M), `smbd`(1M), `smbstat`(1M), `smb`(4), `smbautohome`(4), and `pam_smb_passwd`(5) man pages.

The SMB features offered by the Oracle Solaris service depend on the file system being shared. To fully support the SMB server, 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 Oracle Solaris extended attributes, they are made available as NTFS alternate data streams.
- The case-sensitivity capabilities of the file system are made available to SMB 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 Oracle Solaris OS supports both the NFS and SMB 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. Use mixed mode for maximum multi-protocol compatibility. This mode is enabled by default on ZFS file systems.
- 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 *Oracle Solaris Administration: ZFS File Systems*, respectively.

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

SMB Client

The SMB protocol is the native file-sharing protocol used by Windows and Mac OS systems. The SMB client is an Oracle Solaris virtual file system that provides access to files and directories from the SMB server.

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

For more information about how to use the SMB client to access shares, see Chapter 4, "SMB Client Administration (Tasks)," and the `mount_smbfs`(1M), `smbadm`(1M), `smb`(4), `pam_smbfs_login`(5), and `smbfs`(7FS) man pages.

Identity Mapping Service

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

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

- **Directory-based mapping.** If configured, the `idmap` service tries to use mapping information that is stored in a directory with other user or group information.
  - **Directory-based name mapping.** Uses name mapping information that is stored in user or group objects in the Active Directory (AD), the native LDAP directory service, or both, to map users and groups.
  - **Identity Management for UNIX (IDMU) directory mapping.** Uses UID and GID information stored in the AD data for the Windows user or group. IDMU is an optional AD component that was introduced in Windows Server 2003R2.
- **Rule-based mapping.** Uses rules to map Windows and Oracle 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.

The `idmap` command 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 32. For information about how to determine your identity mapping strategy, see "Creating Your Identity Mapping Strategy" on page 33. For instructions on how to use the `idmap` command, see "Managing Directory-Based Name Mapping for Users and Groups

(Task Map)" on page 36, "Managing Rule-Based Identity Mapping for Users and Groups (Task Map)" on page 48, and the `idmap(1M)` man page.

Managing SMB Configuration Properties

The SMB server and the SMB client use the `sharectl` command to manage configuration properties. For descriptions of the SMB client and server properties, see the `sharectl(1M)` and `smb(4)` man pages.

The SMB properties and their values are stored in the Service Management Facility (SMF). For more information about SMF, see Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*.

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, "SMB Server Administration (Tasks)," and Chapter 4, "SMB Client Administration (Tasks)."

Configuring the SMB Server – Process Overview

This section describes the high-level process for configuring the SMB server.

1. Determine your identity mapping strategy.

   See "Creating Your Identity Mapping Strategy" on page 33.
2. Disable the Samba service.

   See "Disabling the Samba Service" on page 60.
3. Determine whether you want the SMB server to join an existing *Windows domain* or a *Windows workgroup*.
   - To join a domain, see "How to Configure the SMB Server in Domain Mode" on page 61.
   - To join a workgroup, see "How to Configure the SMB Server in Workgroup Mode" on page 63.
4. Define one or more SMB shares.

   See "Managing SMB Shares (Task Map)" on page 65.
5. Configure the Oracle Solaris system as a client of the following services that you might use in your environment.
   - For DNS, see *Oracle Solaris Administration: Naming and Directory Services*.
   - For Kerberos, see "Configuring Kerberos Clients (Task Map)" in *Oracle Solaris Administration: Security Services*.
   - For LDAP, see Chapter 12, "Setting Up LDAP Clients (Tasks)," in *Oracle Solaris Administration: Naming and Directory Services*.

- For NIS, see "Setting Up NIS Clients" in *Oracle Solaris Administration: Naming and Directory Services*.
- For NTP, see "How to Set Up an NTP Client" in *Oracle Solaris Administration: Network Services*.

6. Configure the SMB server as a client to the various services that are used in your environment.
   - For WINS, see "How to Configure WINS" on page 80.

Utilities and Files Associated With the SMB Server and Client

This section describes the SMB utilities and files that are used by the SMB server and client.

- "SMB Utilities" on page 15
- "SMB Service Daemon" on page 18
- "SMB Files" on page 18

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

SMB 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 15
- "`sharectl` Command" on page 16
- "`share` Command" on page 16
- "`smbadm` Command" on page 16
- "`smbstat` Command" on page 17
- "`umount_smbfs` Command" on page 17
- "`unshare` Command" on page 17
- "`zfs` Command" on page 18

`mount_smbfs` Command

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

For more information, see the following:

- "How to Mount an SMB Share on a Directory You Own" on page 87
- "How to Mount a Multiuser SMB Share" on page 92

- "How to Add an Automounter Entry for an SMB Share" on page 94

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

sharectl Command

The sharectl command is an administrative tool that enables you to configure and manage file-sharing protocols, such as SMB 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 command, see the following:

- "How to Configure WINS" on page 80
- "How to Customize the SMB Environment in Oracle Solaris" on page 93
- "How to View the SMB Environment Property Values" on page 94

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

share Command

The share command enables you to manage SMB shares on various file system types. See the share(1M) man page.

You can also use the zfs command to configure SMB sharing on Oracle Solaris ZFS file systems. For more information, see "How to Create an SMB Share (zfs)" on page 67 and the zfs(1M) man page.

For information about SMB share properties, see the share_smb(1M) man page.

smbadm Command

You can use the smbadm command to manage domain membership of the SMB server. You can have the SMB server use domain mode or workgroup mode. The smbadm command also enables you to configure SMB local groups. SMB local groups can be used when Windows accounts must be members of some local groups and when Windows-style privileges must be granted. Oracle Solaris local groups cannot provide this functionality. This command also includes subcommands that enable you to show Windows Server Service information locally on the server.

Use the smbadm command to perform the following SMB client tasks:

- View the shares available for mounting from a particular SMB server
- Generate a hash of a password for storing in a file
- Create or remove persistent passwords used to authenticate to SMB servers

- Resolve a name to an IP address for a server that uses SMB over NetBIOS, not TCP
- Resolve the specified server to the NetBIOS workgroup and system name

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

- "How to Configure the SMB Server in Domain Mode" on page 61
- "How to Configure the SMB Server in Workgroup Mode" on page 63
- "How to Create an SMB Group" on page 77
- "How to Add a Member to an SMB Group" on page 78
- "How to Remove a Member From an SMB Group" on page 78
- "How to Modify SMB Group Properties" on page 79
- "How to Find Available SMB Shares on a Known File Server" on page 86
- "How to Mount an SMB Share on a Directory You Own" on page 87
- "How to Store an SMB Persistent Password" on page 89
- "How to Configure the PAM Module to Store an SMB Persistent Password" on page 90
- "How to Delete an SMB Persistent Password" on page 91
- "How to Mount a Multiuser SMB Share" on page 92

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 SMB server as well as dispatched SMB request counters. For more information, see the `smbstat`(1M) man page.

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

`umount_smbfs` Command

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

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

`unshare` Command

The `unshare` command enables you to remove SMB shares from various file system types. See the `unshare`(1M) man page.

You can also use the `zfs` command to remove SMB shares from a ZFS file system. See "How to Remove an SMB Share (`zfs`)" on page 74.

zfs Command

The zfs command enables you to create, modify, and remove SMB shares on ZFS file systems. See the zfs(1M) man page.

SMB Service Daemon

The smbd daemon and the svc:/network/smb/server service collaborate with the kernel to provide the SMB service. The smb/server service depends on the smb/client service.

The SMB service depends on the idmap service. 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.

SMB Files

The following files support SMB activities on any Oracle Solaris system:

- /etc/auto_direct
- /etc/dfs/sharetab
- /etc/smbautohome

/etc/auto_direct File

Use the /etc/auto_direct file to automatically mount an SMB 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 an SMB Persistent Password" on page 89.

For instructions and examples, see "How to Add an Automounter Entry for an SMB Share" on page 94.

/etc/dfs/sharetab File

The /etc/dfs/sharetab file contains a record of the active shares on the system. Each entry in the file describes a share, which includes the mount point, share name, protocol, and share properties. See the sharetab(4) and share_smb(1M) man pages.

/etc/smbautohome File

The /etc/smbautohome file is used to define the automatic sharing rules to be applied when a user connects to the SMB server. For more information, see "Autohome Shares" on page 22 and the smbautohome(4) man page.

Authentication, Directory, Naming, and Time Services

This section describes the various services that the SMB server interoperates with as a client.

The SMB server interoperates with a variety of naming services that are used by Windows and Oracle 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 Directory 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 80.

SMB Shares

A shared resource, or *share*, is a local resource on a server that is accessible to SMB clients on the network. For the SMB server, a share is typically a directory. Each share is identified by a name on the network. An SMB client sees the share as a complete entity on the SMB 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 SMB server provides a special kind of share called an autohome SMB 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.

Share Properties

You can use share properties to modify the attributes and behavior of an SMB share. Use the `zfs set` and `share` commands to set share properties. There are two types of share properties: global and protocol-specific.

The global share properties include the following:

- `desc` – Specify an optional description of the share
- `name` – Specify the name of the share
- `path` – Specify the mount point of the share
- `prot` – Specify the protocol of the share, such as SMB or NFS

The protocol-specific share properties for the SMB protocol include the following:

- `abe` – Enable or disable access-based enumeration for a share
- `ad-container` – Specify the name of an AD container in which to publish a share
- `catia` – Specify whether to perform CATIA character substitution
- `csc` – Set the client-side caching policy
- `guestok` – Enable or disable guest access to a share
- `ro`, `rw`, `none` – Set host-based access rules for a share

When you specify share properties, specify the global properties first, followed by the `prot` property and then by any protocol-specific properties. For more information about SMB share properties, see the `share_smb`(1M) man page.

To create a share, you *must* specify the `path` property. To change a global share property, specify only the global properties you want to change and not the `prot` property. To change protocol-specific property values, you *must* also specify the `name` and `prot` global share properties.

Access Control to Shares

The SMB server uses the following access-control mechanisms to limit access by users, hosts, or both, to SMB shared file systems (shares):

- **Host-based access control** limits host access to shares.
- **Access control lists (ACLs)** limit user and group access to shares.

Host-based access control is applied first and grants or denies access to the client system. If the host is granted access, the share ACL is applied to grant or deny access to the user. Each mechanism acts as a filter, which might restrict the type of access granted based on the access-control setting.

Shares are always created with the default share ACL and, unless otherwise specified when the share is created, default host-based access control. You can apply non-default values to the share after the share is created.

Host-Based Access Control to Shares

This access-control mechanism enables you to limit the access of a host or group of hosts to an SMB share. This mechanism is a share-level access control and does not apply to local file access. By default, all hosts have full access to a share. The SMB server enforces host-based access control each time a client requests a connection to a share.

You can use the `zfs set` and `share` commands to specify host-based access control on a share. For more information, see "How to Restrict Client Host Access to an SMB Share (`zfs`)" on page 75, and the `share`(1M) and `zfs`(1M) man pages.

Access Control Lists on Shares

An ACL on a ZFS share provides the same level of access control as a Windows ACL does for its shares. Each share can have an ACL that includes entries to specify which types of access are allowed or denied to users and groups. Like host-based access control, this mechanism is a share-level form of access control and does not apply to local file access.

These share ACLs are only available for ZFS shares. You can manage a ZFS share's ACL in the Oracle Solaris OS by using the `chmod` and `ls` commands. See the `chmod`(1) and `ls`(1) man pages. You can also manage these ACLs by using the Windows share management GUI on a Windows client.

Although a ZFS file system is used to store a share's ACL, the access control is enforced by the SMB server each time a client requests a connection to a share. The default ACL setting permits full access to everyone.

Note – You *cannot* specify an ACL on an autohome share. Autohome shares are created at runtime with a predefined, unmodifiable ACL that grants full control to the owner. Only the autohome share owner can access the share.

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.

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 74.

When a user logs in, the SMB server looks for a subdirectory that matches the user's name based on any rules that have been specified. If the server 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 server 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 SMB autohome shares. Even after an SMB 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 SMB server is restarted.

Autohome Entries

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

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

```
+nsswitch        dc=ads,dc=oracle,dc=com,ou=users
jane    /home/?/&    dc=ads,dc=oracle,dc=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 Directory 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 prefixes:

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

`cn=`, `ou=`, and `dc=` are attribute types. The attribute type used to describe an object's RDN is called the naming attribute, which for AD includes the following object classes:

- `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 (&) 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 (`?`) 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 (*) 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 SMB Groups

Local SMB groups can be created on the system that runs the SMB server. These SMB groups apply only to users that are connected through SMB.

The SMB server supports the following built-in SMB 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 SMB groups, see “Managing SMB Groups (Task Map)” on page 76.

Client-Side Caching for Offline Files

The SMB server provides a per-share configuration property to support client-side caching for offline files. Although the SMB server enables you to configure this feature, only the client manages client-side caching and access to offline files. You can use the `zfs` command to configure this feature by setting the `csc` property for a share.

The following are valid values for the csc property:

- manual – Permits clients to cache files from the specified share for offline use as requested by users. However, automatic file-by-file reintegration is not permitted. manual is the default value.
- auto – Permits clients to automatically cache files from the specified share for offline use, and permits file-by-file reintegration.
- vdo – Permits clients to automatically cache files from the specified share for offline use, permits file-by-file reintegration, and permits clients to work from their local cache even while offline.
- disabled – Disables client-side caching for the specified share.

EXAMPLE 1–1 Configuring Client-Side Caching

The following example shows how to configure client-side caching on shares.

First, create and share a file system. When you are using SMB, it is best practice to create a mixed-mode ZFS file system, which is the default. If you have both NFS and SMB clients using a mixture of different character sets on the same file system, you might also want to set the utf8only property and consider specifying the charset=*access-list* NFS share property.

The sharesmb property can only be set to on or off. Specifying sharesmb=on during dataset creation shares the dataset with the default share properties.

```
# zfs create -o utf8only=on -o sharesmb=on tank/zvol
# zfs get share tank/zvol

NAME PROPERTY VALUE SOURCE
tank/zvol share name=tank_zvol,path=/tank/zvol,prot=smb local
```

If you specify sharesmb=on during dataset creation, the share is automatically created as a default share. The name of the share is based on the share path, where slashes (/) are replaced by underscores (_).

To create a share with non-default values, use the zfs set share command, as shown in the following example:

First, create the dataset.

```
# zfs create -o utf8only=on tank/zvol
```

Next, create an SMB share with the name of zvol.

```
# zfs set share=name=zvol,path=/tank/zvol,prot=smb tank/zvol
name=zvol,path=/tank/zvol,prot=smb

# zfs get share tank/zvol
NAME PROPERTY VALUE SOURCE
```

EXAMPLE 1–1 Configuring Client-Side Caching *(Continued)*

```
tank/zvol share name=zvol,path=/tank/zvol,prot=smb local
```

Then, enable SMB sharing on the `tank/zvol` dataset, and view the active shares on the system.

```
# zfs set sharesmb=on tank/zvol
# cat /etc/dfs/sharetab
/tank/zvol zvol smb -
```

The following command creates a new share, `zvol2`, with the `csc` property set to `auto`:

```
# zfs set share=name=zvol2,path=/tank/zvol2,prot=smb,csc=auto tank/zvol2
name=zvol2,path=/tank/zvol2,prot=smb,csc=auto

# zfs get share tank/zvol2
NAME PROPERTY VALUE SOURCE
tank/zvol2 share name=zvol2,path=/tank/zvol2,prot=smb,csc=auto local
```

Using the `zfs` command enables you to add properties to a share without specifying all the other previously specified properties and their values.

In the following example, the first command creates a share with the name of `zvol3`. The second command adds the `csc` property. In the second command, you do not need to specify the `path` property because it was already specified in the first command.

```
# zfs set share=name=zvol3,path=/tank/zvol3,prot=smb tank/zvol3
name=zvol3,path=/tank/zvol3,prot=smb

# zfs set share=name=zvol3,prot=smb,csc=auto tank/zvol3
name=zvol3,prot=smb,csc=auto
```

To add an SMB property with the `zfs` command, specify the share name (`name=zvol3`), the protocol (`prot=smb`), and the new property (`csc=auto`).

You can also set the `csc` property on autohome shares in the `smbautohome` map. As with the ZFS `share` property, multiple property-value pairs can be specified in a comma-separated list. The following `smbautohome` map disables client-side caching by default, but sets `csc=auto` for `/export/home/john`:

```
*      /export/home/&   csc=disabled,description=&
john   /export/home/&   csc=auto,dn=oracle,dn=com,ou=users
```

Share Execution Properties

The SMB server provides a set of service properties to support the execution of a command or script when SMB shares are connected or disconnected. These properties are configurable with the `sharectl` command and are applied to all shares. You can use the command or script to perform automated administrative tasks each time a share is mapped (connected) or unmapped (disconnected). These scripts and commands must be run as superuser. For example, you might use a command to create home directories or to monitor resources.

You must be superuser or assume an equivalent role to obtain the `solaris.smf.modify.application` RBAC authorization to use `sharectl` to configure these properties.

The service property names and values are as follows:

- `map`. The value of this property is a command to be executed when the client connects to the share. The command can take the following arguments, which are substituted when the command is executed by `exec()`:
  - `%D` – Domain or workgroup name of `%U`.
  - `%h` – Server host name.
  - `%I` – IP address of the client system.
  - `%i` – Local IP address to which the client is connected.
  - `%L` – Server NetBIOS name.
  - `%M` – Client host name, or `“”` if not available.
  - `%m` – Client NetBIOS name, or `“”` if not available. This option is only valid for NetBIOS connections (port 139).
  - `%P` – Root directory of the share.
  - `%S` – Share name.
  - `%U` – Windows user name.
  - `%u` – UID of the UNIX user.
- `unmap`. The value of this property is a command to be executed when the client disconnects from the share. The command can use the same arguments that are described for the `map` property.
- `disposition=[continue|terminate]`. This property controls whether to disconnect the share or proceed if the `map` command fails. This property only has meaning when the `map` property has been set. Otherwise, it has no effect.

The following are valid values for the disposition property:

- continue – Proceed with the share connection if the map command fails. This is the default behavior when the disposition property is not specified.
- terminate – Disconnect the share if the map command fails.

EXAMPLE 1–2 Using Share Execution Properties

The following sharectl examples show how you might set the map, unmap, and disposition properties:

```
# sharectl set -p map="/tmp/map_script %U" smb
# sharectl set -p unmap=/tmp/unmap_script smb
# sharectl set -p disposition=terminate smb
```

The first command runs the /tmp/map_script *Windows-username* command when a share is mapped. The second command runs the /tmp/unmap_script command when a share is unmapped. The third command specifies that the share will disconnect if the command fails during the mapping operation.

Support for the Distributed File System

The Distributed File System (DFS) feature is supported by the SMB server. For more information, see the Microsoft DFS documentation.

Currently, the SMB server supports only *one* stand-alone DFS namespace per system.

No configuration is required on the SMB server to use the DFS feature. You can use the DFS tools that are available on Windows systems to create and manage the stand-alone namespace in the Oracle Solaris OS.

Support for SMB Printing

SMB printing enables you to gain remote access to all of the local Common UNIX Printing System (CUPS) printers. See “Configuring SMB Printing (Task Map)” on page 82.

◆ ◆ ◆ CHAPTER 2

Identity Mapping Administration (Tasks)

This chapter describes the identity mapping service that maps Windows security identifiers (SIDs) to Oracle 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 `idmap` service can run in the global zone or in non-global zones. However, if Oracle Solaris Trusted Extensions software is enabled, the `idmap` service *must* run in the global zone.

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

Up-to-date troubleshooting information is available on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

Mapping User and Group Identities

The SMB server is designed to reside in a multiprotocol environment and provide an integrated model for sharing data between Windows and Oracle Solaris systems. Although files can be accessed simultaneously from both Windows and Oracle Solaris systems, no industry-standard mechanism is available to define a user in both Windows and Oracle Solaris environments. Objects can be created in either environment, but traditionally the access control semantics for each environment are vastly different. The Oracle 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 `idmap` identity mapping service.

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

The SMB server determines the Windows user's Oracle Solaris credentials by using the `idmap` 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 Oracle 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 Oracle Solaris system reboots. Ephemeral mappings enable the SMB server to work in a Windows environment without having to configure any name-based mappings.

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

- **Directory-based mapping.** If configured, `idmap` first tries to use mapping information that is stored in a directory with other user and group information.
  - **Directory-based name mapping.** In this mode, `idmap` 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 Oracle Solaris user or group name. Similarly, the native LDAP object for a particular Oracle Solaris user or group can be augmented to include the corresponding Windows user or group name.

    You can configure `idmap` to use AD, native LDAP directory-based name mappings, or both, by setting the `idmap` service properties in the Service Management Facility (SMF). See Service Properties in the `idmap(1M)` man page.
  - **Identity Management for UNIX (IDMU).** In this mode, `idmap` tries to use UID or GID information that is stored in the AD data for the Windows user or group. IDMU is an optional AD component that was added to Windows Server 2003R2. IDMU adds a UNIX Attributes tab to the Active Directory Users and Computers user interface.

If directory-based name mapping is not configured, or if it is configured but the user or group entry does not include mapping data, `idmap` will continue to try additional mapping mechanisms.

- **Rule-based mapping.** This mechanism allows the administrator to define rules that associate Windows and Oracle Solaris users and groups by name.
- **Ephemeral ID mapping.** Windows users and groups that have no corresponding Oracle Solaris user or group are assigned temporary UIDs and GIDs. Over two billion identifiers are available for use. This mechanism is largely transparent if you have the `ad` source configured for the `passwd` and `group` databases in SMF. For more information, see Chapter 16, "Setting Up Oracle Solaris Active Directory Clients," in *Oracle Solaris Administration: Naming and Directory Services*.

You can use the `idmap` command to create and manage the rule-based mappings. These rules map the specified Windows name to the specified Oracle Solaris name, and map the specified Oracle 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 Oracle Solaris user. Note that `dana@example.com` maps to `danas`, and `danas` maps to `dana@example.com`.

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

For more information about other mapping types, see the `idmap(1M)` man page.

Creating Your Identity Mapping Strategy

Your SMB server can use directory-based mapping, rule-based mapping, both, or neither. By default, Windows users and groups do not need to be associated with Oracle Solaris users and groups. Without any mapping, Windows users and groups can still own files, be listed in ACLs, and such. Identity mapping is required when users need access to files from both Windows and Oracle Solaris operating systems or NFS. These mappings enable a user to be treated the same whether locally logged in or connected from a Windows system or through NFS.

If your Windows environment includes a parallel Oracle Solaris naming service infrastructure, such as NIS, consider using *name-based mappings* to associate Windows users with Oracle Solaris users, and Windows groups with Oracle Solaris groups.

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.

- **Using directory-based name mapping.** Directory-based name 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 SMB 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 service or services to employ:

  - If you have already deployed AD or native LDAP, use that naming service.
  - For 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.

  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 38 and "How to Extend the Native LDAP Schema, and User and Group Entries" on page 40.

     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 and to inform the `idmap` service about the attributes to be used. See "How to Configure Directory-Based Mapping" on page 42.

- **Using Identity Management for UNIX.** IDMU is an optional Active Directory feature that enables administrators to specify UNIX-specific information for Active Directory users and groups. When IDMU support is enabled, `idmap` uses the UID and GID information maintained by IDMU to map Windows users and groups to the equivalent Oracle Solaris users and groups. Use IDMU in the following situations:

  - You want to use a user interface that is integrated into the Active Directory user interface.
  - You are using IDMU and a Windows NIS server to provide UNIX naming services.

IDMU data is used only for users and groups in the domain to which the Oracle Solaris system is joined. If it is necessary to provide mappings for users and groups from other domains, you must use a different strategy, either in addition to or instead of IDMU.

To use IDMU, do the following:

1. Use the Windows Server Manager to install IDMU on the Active Directory domain controller.
2. Use the UNIX Attributes tab in the Active Directory Users and Computers tool to specify UIDs and GIDs for your users.
3. Enable `idmap` IDMU support. See "How to Enable Identity Management for UNIX Support" on page 47.

- **Using rule-based mapping.** A *rule-based mapping* uses rules to associate Windows users and groups with equivalent Oracle Solaris users and groups by name rather than by identifier.

  These mappings are easy to configure and can be configured with a single wildcard rule. However, the mapping rules are only stored on a particular system rather than being global. This method is more suitable if only one SMB server is being used in your environment.

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

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

     The first command maps the Windows user called `pat@example.com` to the Oracle Solaris user `pat`. The second command maps the Windows group called `staff@example.com` to the Oracle Solaris group `staff`.

     Note – You can only have one bidirectional rule-based mapping to map all users in a single Windows domain to all Oracle Solaris users in the local Oracle Solaris domain. If you instead had wildcard mappings for two domains, it would not be possible to determine which domain to use when mapping an Oracle Solaris user to a Windows user.

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

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

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

Mapping Well-Known Account Names

The idmap service supports the mapping of well-known Windows account names, such as the following:

- Administrator
- Guest
- Network
- Administrators
- Guests
- Computers

When idmap rules are added, these well-known account names are expanded to canonical form, which adds either the default domain name (for names that are not well known) or an appropriate built-in domain name. Depending on the particular well-known name, this domain name might be null, BUILTIN, or the local host name.

The following sequence of idmap commands shows the treatment of the name dana, which is not well known, and the well-known names administrator and guest:

```
# idmap add winname:dana unixuser:danam
add     winname:dana    unixuser:danam
# idmap add winname:administrator unixuser:root
add     winname:administrator   unixuser:root
# idmap add winname:guest unixuser:nobody
add     winname:guest   unixuser:nobody
# idmap add wingroup:administrators sysadmin
add     wingroup:administrators unixgroup:sysadmin
# idmap list
add     winname:Administrator@examplehost  unixuser:root
add     winname:Guest@examplehost  unixuser:nobody
add     wingroup:Administrators@BUILTIN unixgroup:sysadmin
add     winname:dana@example.com       unixuser:danam
```

Managing Directory-Based Name 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 SMB server 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 38
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 40
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 42
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 44
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 45
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 46
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 46

For more information about user and group identities, see "Mapping User and Group Identities" on page 32. For more information about how to determine your identity mapping strategy, see "Creating Your Identity Mapping Strategy" on page 33.

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 Oracle Solaris names.

Note – Perform this task before enabling directory-based mapping on your Oracle 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/en-us/default.aspx`).

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 also 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 44 and "How to Add a Directory-Based Name Mapping to a Group Object" on page 45.

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 Oracle Solaris users `tmw`, `crj`, and `dab`, respectively.

This example shows how to add the Oracle 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 Oracle 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 Oracle 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 44 and "How to Add a Directory-Based Name Mapping to a Group Object" on page 45.

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 Oracle Solaris users tmw, crj, and dab stored in native LDAP. These Oracle Solaris users are associated with the Windows users terry, cal, and dana, respectively, all in the domain example.com.

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 Oracle 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 Oracle Solaris system, you must extend the AD schema, the native LDAP schema, or both, and populate the user and group objects with the associated Oracle Solaris names. See “How to Extend the Active Directory Schema, and User and Group Entries” on page 38 and “How to Extend the Native LDAP Schema, and User and Group Entries” on page 40.

1 **Enable directory-based mapping.**

```
# svccfg -s svc:/system/idmap setprop config/directory_based_mapping=astring: name
```

The `directory_based_mapping` property controls support for identity mapping that uses data stored in a directory service. The value of the `directory_based_mapping` property can be one of the following:

- `none` – Disables directory-based mapping.
- `name` – Enables name-based mapping by using the `config/ad_unixuser_attr`, `config/ad_unixgroup_attr`, and `config/nldap_winname_attr` properties. These properties are described in the `idmap(1M)` man page.
- `idmu` – Enables mapping by using Identity Management for UNIX (IDMU). IDMU is a Windows component that permits the administrator to specify a UNIX user ID for each Windows user, and then have the Windows identity mapped to the corresponding UNIX identity. Note that only IDMU data is used from the domain of which the Oracle Solaris system is a member.

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
  ```

3 **Refresh the identity mapping service.**

```
# svcadm refresh svc:/system/idmap
```

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

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

1 **Become an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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 Oracle Solaris user `dana` by adding the Oracle Solaris name to the AD object for `danab@example.com`:

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

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

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

  For example, the following command maps Oracle 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 an Oracle Solaris group by adding the Oracle Solaris group name to the AD object for the specified Windows group.
- Map an Oracle Solaris group to a Windows group by adding the Windows group name to the native LDAP object for the specified Oracle Solaris group.

1 **Become an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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 Oracle Solaris group `sales` by adding the Oracle Solaris name to the AD object for `salesgrp@example.com`:

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

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

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

For example, the following command maps Oracle 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 an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.

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

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 Oracle Solaris name from the AD object for the specified user.**

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

  For example, the following command removes the Oracle 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 Oracle Solaris user `dana`:

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

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

1 Become an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.

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

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 Oracle 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 Oracle 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 Oracle Solaris group `sales`:

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

Managing Directory-Based Identity Mapping by Using Identity Management for UNIX (Task Map)

You can use the following task to enable Identity Management for UNIX (IDMU) to manage directory-based identity mapping for the SMB server in a Windows environment. IDMU is an optional feature of Active Directory.

▼ How to Enable Identity Management for UNIX Support

Before You Begin Before you can use IDMU support, you must first install the IDMU software on your Active Directory domain controller and use the UNIX Attributes tab in the Active Directory Users and Computers tool to specify UIDs and GIDs for your users.

1 **Become an administrator.**

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

2 **Enable IDMU support.**

```
# svccfg -s svc:/system/idmap setprop \
config/directory_based_mapping = astring: idmu
```

3 **Refresh the identity mapping service.**

```
# svcadm refresh svc:/system/idmap
```

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

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

By default, no rule-based mappings are configured. In this case, non-ephemeral Oracle 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 SMB server 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 Oracle Solaris users based on the names in the naming services.	"How to Add a User Mapping Rule" on page 49
Add a group mapping rule.	Use rules to create identity equivalents for Windows groups and Oracle Solaris groups based on the names in the naming services.	"How to Add a Group Mapping Rule" on page 51
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 53
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 54

Task	Description	For Instructions
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 54
Show all the established mappings.	Use this procedure to view the mappings stored in the cache.	"How to Show All Established Mappings" on page 55
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 56
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 57

For more information about user and group identities, see "Mapping User and Group Identities" on page 32. For more information about how to determine your identity mapping strategy, see "Creating Your Identity Mapping Strategy" on page 33.

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 Oracle Solaris users. By default, the SMB server 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 an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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 an Oracle 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 Oracle Solaris user that you want to map to the Windows user.**

The Oracle 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. Do *not* use a user name of "" 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 Oracle Solaris name and the Oracle 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 Oracle Solaris users. For example, the `'*@example.com' == '*'` rule ensures that the `jp@example.com` Windows user name maps to the `jp` Oracle 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 Oracle 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.

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

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

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

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

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

- **Create a unidirectional mapping between an Oracle 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 Oracle Solaris groups. By default, the SMB server uses ephemeral identity mapping.

You can also create diagonal mappings to maps between a Windows group and an Oracle Solaris user and between an Oracle 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 an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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 an Oracle 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 Oracle Solaris user or group that you want to map to the Windows group.**

  The Oracle Solaris group name must be specified by using the format `unixgroup:`*group-name*. The Oracle 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 Oracle Solaris group name, and the Oracle 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 Oracle 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` Oracle 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 Oracle 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.

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

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

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

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

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

- **Create a unidirectional mapping between an Oracle 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 an Oracle Solaris user name.**

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

- **Create a diagonal mapping between an Oracle 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 an Oracle 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 an Oracle 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 an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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 2147926017, type:

```
$ idmap show uid:2147926017 sid
uid:2147926017 -> sid:S-1-5-21-721821396-1083305290-3049112724-500
```

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

```
$ idmap show administrator@example.com
winuser:administrator@example.com -> uid:2147926017
```

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. The `-n` option shows names instead of IDs.

```
$ idmap dump -n
winuser:dana@a.terry.example.com <= uid:2147909633
winuser:u2@a.terry.example.com <= uid:2147909634
wingroup:Group Policy Creator Owners@a.terry.example.com == gid:2147917831
wingroup:Domain Admins@a.terry.example.com == gid:2147917832
wingroup:Enterprise Admins@a.terry.example.com == gid:2147917833
wingroup:Schema Admins@a.terry.example.com == gid:2147917834
wingroup:Netmon Users@a.terry.example.com == gid:2147917836
wingroup:Administrators@BUILTIN == gid:2147917837
usid:S-1-5-21-156362980-169493972-3399456007-500 == uid:2147917825
usid:S-1-5-21-156362980-169493972-3399456007-520 == gid:2147917826
usid:S-1-5-21-156362980-169493972-3399456007-512 == gid:2147917827
```

```
usid:S-1-5-21-156362980-169493972-3399456007-519 == gid:2147917828
usid:S-1-5-21-156362980-169493972-3399456007-518 == gid:2147917829
wingroup:Network == gid:2147557379
wingroup:Authenticated Users == gid:2147917830
winuser:administrator@solar == uid:2147926017
winuser:Administrator@a.terry.example.com == uid:2147557377
usid:S-1-5-21-156362980-169493972-3399456007-513 == gid:2147557378
```

- To optionally list only the user mappings, type:

  ```
  $ idmap dump -n | grep uid
  winuser:dana@a.terry.example.com <= uid:2147909633
  winuser:u2@a.terry.example.com <= uid:2147909634
  usid:S-1-5-21-156362980-169493972-3399456007-500 == uid:2147917825
  winuser:administrator@solar == uid:2147926017
  winuser:Administrator@a.terry.example.com == uid:2147557377
  ```

- To optionally list only the group mappings, type:

  ```
  $ idmap dump -n | grep gid
  wingroup:Group Policy Creator Owners@a.terry.example.com == gid:2147917831
  wingroup:Domain Admins@a.terry.example.com == gid:2147917832
  wingroup:Enterprise Admins@a.terry.example.com == gid:2147917833
  wingroup:Schema Admins@a.terry.example.com == gid:2147917834
  wingroup:Netmon Users@a.terry.example.com == gid:2147917836
  wingroup:Administrators@BUILTIN == gid:2147917837
  usid:S-1-5-21-156362980-169493972-3399456007-520 == gid:2147917826
  usid:S-1-5-21-156362980-169493972-3399456007-512 == gid:2147917827
  usid:S-1-5-21-156362980-169493972-3399456007-519 == gid:2147917828
  usid:S-1-5-21-156362980-169493972-3399456007-518 == gid:2147917829
  wingroup:Network == gid:2147557379
  wingroup:Authenticated Users == gid:2147917830
  usid:S-1-5-21-156362980-169493972-3399456007-513 == gid:2147557378
  ```

▼ How to Remove a User Mapping Rule

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

1 **Become an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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

```
# idmap list
```

For example, to find all user mappings that map to the Oracle 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 an administrator, obtain the `solaris.admin.idmap.rules` RBAC authorization, or use the Idmap Service Management RBAC profile.**

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

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

```
# idmap list
```

For example, to find all unidirectional group mappings that map to the Oracle 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

SMB Server Administration (Tasks)

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

Currently, the SMB service runs only in the global zone.

This chapter covers the following topics:

- "Disabling the Samba Service" on page 60
- "Configuring the SMB Server Operation Mode (Task Map)" on page 60
- "Managing SMB Shares" on page 64
- "Managing SMB Groups (Task Map)" on page 76
- "Configuring the WINS Service" on page 80
- "Enabling CATIA V4/V5 Character Translations" on page 80
- "Configuring SMB Printing (Task Map)" on page 82

For a high-level overview of the SMB server configuration process, see "Configuring the SMB Server – Process Overview" on page 14.

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

Up-to-date troubleshooting information is available on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

For information about installing the SMB server packages, see Getting Started With the Oracle Solaris SMB Service wiki on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

Disabling the Samba Service

Samba and SMB servers cannot be used together on a single Oracle Solaris system. To run the SMB server, you must first ensure that a running Samba service is disabled.

If your Oracle Solaris system is running the Samba service, disable it before starting the SMB server.

▼ How to Disable the Samba Service

1. **Become an administrator.**

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

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
   ```

Configuring the SMB Server Operation Mode (Task Map)

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

Task	Description	For Instructions
Configure the SMB server in domain mode.	Use the `smbadm join -u` *username domain-name* command to join the domain.	"How to Configure the SMB Server in Domain Mode" on page 61
Configure the SMB server in workgroup mode.	Use the `smbadm join -w` *workgroup-name* command to join the workgroup.	"How to Configure the SMB Server in Workgroup Mode" on page 63

▼ How to Configure the SMB Server in Domain Mode

This procedure describes how to use the `smbadm join` command to join an AD domain. To instead use the `kclient` command to manually join the domain, see "How to Configure a Kerberos Client for an Active Directory Server" in *Oracle Solaris Administration: Security Services*.

After successfully joining an AD domain, you can enable the SMB server to publish SMB shares in the AD directory. To do so, create or update SMB shares and specify the share container for each share that you want to publish. To create SMB shares, see "How to Create an SMB Share (`zfs`)" on page 67.

Starting with the Oracle Solaris 11 OS, the `smbadm join` command automatically configures Kerberos. If you are running a version of the Solaris Express OS or the Oracle Solaris 11 Express OS, you must manually configure Kerberos as described in the following Before You Begin section.

Before You Begin If the Samba service is running on the Oracle Solaris system, you must disable it. See "How to Disable the Samba Service" on page 60.

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 SMB server 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.

To participate in an AD domain, the system must be configured to use DNS for host lookup. Ensure that the naming service and the DNS service are configured correctly for the appropriate AD domain.

If you are running a version of the Solaris Express OS or the Oracle Solaris 11 Express OS, you must manually configure Kerberos as described in the following paragraphs.

In the `/etc/krb5/krb5.conf` file, specify the fully qualified AD domain name, in uppercase letters, 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 that has multiple AD domain controllers. The primary AD domain controller is called dc.example.com. A secondary AD domain controller is called dc2.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
       kdc = dc2.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 example file, see the krb5.conf(4) man page and "Configuring Kerberos Clients (Task Map)" in *Oracle Solaris Administration: Security Services*.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Enable the SMB 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 Oracle 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 Oracle Solaris system or the DC to match the other.**
- **Configure both the Oracle Solaris system and the DC to use the same time source (NTP server).**
- **Synchronize the system clock on the Oracle Solaris system with the system clock of the DC by running the following command on the Oracle 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 SMB Server in Domain Mode

This example shows the steps taken to configure the SMB server 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
After joining westsales.example.com the smb service will be restarted automatically.
Would you like to continue? [no]:
Enter domain password:
Joining 'westsales.example.com' ... this may take a minute ...
Successfully joined domain 'westsales.example.com'
```

▼ How to Configure the SMB Server in Workgroup Mode

To create SMB shares, see "How to Create an SMB Share (`zfs`)" on page 67.

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

Before You Begin If the Samba service is running on the Oracle Solaris system, you must disable it. See "How to Disable the Samba Service" on page 60.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Enable the SMB service.**

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

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

3 **(Optional) Change the SMB server to operate in a different workgroup.**

By default, the SMB server 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 SMB.**

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 SMB server cannot use the Oracle 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 SMB server 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 SMB Server in Workgroup Mode

This example shows how to configure the SMB server in workgroup mode. The name of the workgroup being joined is myworkgroup.

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

Then, create a share. See “How to Create an SMB Share (zfs)” on page 67.

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

```
# passwd cal
```

Now, you are ready to have SMB clients access the SMB shares on your SMB server.

Managing SMB Shares

You can add, view, and update SMB shares. A directory must exist before it can be shared. For more information about SMB shares, see “SMB Shares” on page 19.

Managing SMB Shares in This Release

The Oracle Solaris 11 OS introduces a new method for sharing and managing SMB and NFS shares. The zfs command has been enhanced to manage shares and share properties on Oracle Solaris ZFS file systems. The zfs command now supports SMB and NFS sharing by means of the share, sharesmb, and sharenfs properties.

The legacy sharemgr command is no longer available to manage SMB shares. Instead, use the enhanced zfs, share, and unshare commands. Also, the automatic sharing of SMB and NFS shares is managed by SMF rather than by the legacy /etc/dfs/dfstab file, which has been removed.

You can continue to use the legacy file-sharing method to manage shares on file servers that run previous versions of the Oracle Solaris OS. For information about the differences between the new and legacy file-sharing methods, see "New ZFS Sharing and Legacy Share Command Summary" in *Oracle Solaris Administration: ZFS File Systems*.

Managing SMB Shares (Task Map)

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

Task	Description	For Instructions
Enable cross-protocol locking.	Use the mount or the zfs create command to enable cross-protocol locking. These commands enable this locking by setting the nbmand option.	"How to Enable Cross-Protocol Locking" on page 66
Create an SMB share by using the ZFS file system's share property.	Use this procedure to make a dataset available to clients.	"How to Create an SMB Share (zfs)" on page 67
Enable guest access to an SMB share.	Use the zfs command to enable guest access for a specified share. These commands enable this feature by setting the guestok property.	"How to Enable Guest Access to an SMB Share" on page 71
Enable access-based enumeration (ABE) for an SMB share.	Use the zfs command to enable ABE filtering for a specified share. These commands enable this feature by setting the abe property to true.	"How to Enable Access-Based Enumeration for a Share" on page 72
Modify the properties of an SMB share by using the share command.	Use this procedure to change share property values.	"How to Modify SMB Share Properties (zfs)" on page 73
Remove an SMB share by using the unshare 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 an SMB Share (zfs)" on page 74

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 74
Restrict host access to a share by using the ZFS file system `share` property.	Use this procedure 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 an SMB Share (`zfs`)" on page 75

▼ How to Enable Cross-Protocol Locking

The SMB protocol assumes mandatory locking, but UNIX traditionally uses advisory locking. The Oracle 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 SMB server will enforce mandatory share reservations and byte-range locking internally for all SMB clients. However, without `nbmand` set, there is only limited coordination with NFS and local processes.

1 **Become an administrator.**

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

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
  ```

Note – The `casesensitivity` property is set to `mixed` by default on ZFS file systems.

▼ How to Create an SMB Share (`zfs`)

This procedure describes how to use the ZFS file system's `share` property to create ZFS shares on the SMB server.

You can also use the `share` command to create shares on various file system types. See the `share`(1M) man page.

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 74.

1. **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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

   By default, ZFS file systems enable mixed-case mode.

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

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

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

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

   The `sharesmb` property *must* be set to `on` to enable SMB sharing on the dataset.

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

Note – The `zfs` command automatically constructs the default share name in the following circumstances:

- When you create the dataset and set the `sharesmb` property to `on`
- When you create a share without specifying a `name` property value

The share name is based on the name of the dataset mount point. Any characters that are illegal for share names are replaced by an underscore (_).

4 **(Optional) Create an SMB share that has non-default property values or an SMB share for a directory other than the mount point of the dataset.**

Use the `zfs` command to set the `share` property, which is used to create one or more shares per dataset. The `share` property value is a comma-separated list of name-value pairs that define a share. See the `zfs`(1M) man page.

The shares are stored in the `.zfs/shares` directory of the dataset's mount point.

Use the `ls` command to show the share-level ACLs on these entries. 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.

For example, you must specify at least the `name`, `path`, and `protocol` properties to create a share:

```
# zfs set share=name=myshare,path=/mntpnt/directory,prot=smb pool-name/fsname
```

5 **(Optional) Specify additional SMB share properties.**

For more information about SMB share properties, see "Share Properties" on page 20, and the `share_smb`(1M), `share`(1M), and `zfs`(1M) man pages.

The following command creates a new share with the client-side caching policy set to `auto`:

```
# zfs set share=name=smb_share,path=/mntpnt/dir2,prot=smb,csc=auto tank/home
name=smb_share,path=/mntpnt/dir2,prot=smb,csc=auto
```

You can also add properties to existing shares. The following command sets the guest access policy of the share that was created by the previous command to `true`:

```
# zfs set share=name=smb_share,prot=smb,guestok=true tank/home
name=smb_share,path=/mntpnt/dir2,prot=smb,csc=auto,guestok=true
```

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

The `/etc/dfs/sharetab` file contains information about all active shares on the system.

```
# cat /etc/dfs/sharetab
/admins ashare smb csc=auto,guestok=true
```

Example 3–3 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 SMB sharing.

```
# zpool create sandbox -o sharesmb=on c0t3d0
# zfs create -o nbmand=on sandbox/fs1
# zfs create -o nbmand=on sandbox/fs2
# zfs get -r share sandbox

NAME PROPERTY VALUE SOURCE
sandbox share name=sandbox,path=/sandbox,prot=smb local
sandbox/fs1 share name=sandbox_fs1,path=/sandbox/fs1,prot=smb local
sandbox/fs2 share name=sandbox_fs2,path=/sandbox/fs2,prot=smb local
```

Example 3–4 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 and SMB sharing:

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

The ZFS file system constructs the share name based on the dataset mount point when the share is created by setting sharesmb=on. Any illegal characters in the share name are replaced by an underscore (_). In this example, the share name sandbox_fs1 is based on the dataset mount point sandbox/fs1.

The zfs get share command lists all shares that are defined on a mounted file system.

```
# zfs get share sandbox/fs1
NAME PROPERTY VALUE SOURCE
sandbox/fs1 share name=sandbox_fs1,path=/sandbox/fs1,prot=smb local
```

You can also view the list of active shares on the system from the /etc/dfs/sharetab file.

The following commands create another file system in the sandbox pool called fs2, associate the file system with the myshare share name, and enable SMB sharing:

```
# zfs create -o nbmand=on sandbox/fs2
# zfs set share=name=myshare,path=/sandbox/fs2,prot=smb sandbox/fs2
name=myshare,path=/sandbox/fs2,prot=smb
# zfs set sharesmb=on sandbox/fs2
```

You can use the zfs get command to view the sharesmb and share property values for the sandbox pool.

```
# zfs get -r sharesmb sandbox
NAME PROPERTY VALUE SOURCE
sandbox sharesmb off default
sandbox/fs1 sharesmb on local
sandbox/fs2 sharesmb on local

# zfs get -r share sandbox
NAME PROPERTY VALUE SOURCE
sandbox/fs1 share name=sandbox_fs1,path=/sandbox/fs1,prot=smb local
sandbox/fs2 share name=myshare,path=/sandbox/fs2,prot=smb local
```

You can also see the list of all active shares on the system by viewing the /etc/dfs/sharetab file.

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

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

The new file system inherits the sharesmb property from its parent, sandbox/fs2, which causes a new default share to be created.

```
# zfs get share sandbox/fs2/fs2_sub1
NAME PROPERTY VALUE SOURCE
sandbox/fs2/fs2_sub1 share name=sandbox_fs2_fs2_sub1,
path=/sandbox/fs2/fs2_sub1,prot=smb local
```

You can also see the list of all active shares on the system by viewing the /etc/dfs/sharetab file.

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

```
# zfs set sharesmb=off sandbox/fs2
# zfs get -r sharesmb sandbox/fs2
NAME                 PROPERTY   VALUE SOURCE
sandbox/fs2          sharesmb  off local
sandbox/fs2/fs2_sub1 sharesmb  off inherited from sandbox/fs2
```

Note that disabling the sharesmb property only unpublishes the shares but does not remove the share definitions. The /etc/dfs/sharetab file shows that only the sandbox_fs1 share is still published, while the myshare and sandbox_fs2_fs2_sub1 shares still exist but are no longer published.

```
# cat /etc/dfs/sharetab
/sandbox/fs1 sandbox_fs1 smb -
# zfs get -r share sandbox
NAME                 PROPERTY VALUE       SOURCE
sandbox/fs1          share    name=sandbox_fs1,path=/sandbox/fs1,prot=smb local
sandbox/fs2          share    name=myshare,path=/sandbox/fs2,prot=smb local
sandbox/fs2/fs2_sub1 share    name=sandbox_fs2_fs2_sub1,
                              path=/sandbox/fs2/fs2_sub1,prot=smb local
```

Example 3–5 Using ls and chmod to Manage SMB Share-Level ACLs

The following example shows how to view the share-level ACLs on SMB 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.

This example shows how you can manage share ACLs on an Oracle Solaris system. However, it is best practice to use Windows utilities to manage share ACLs.

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 Enable Guest Access to an SMB Share

When you have guest access to a share, you are permitted access to the share even if you are not a regular user of the system. You do not need to present credentials for authentication to gain access to that share.

The SMB server uses the `guestok` share property to specify whether guest access is permitted for a given share. If `guestok` is set to `true`, guest access is enabled. However, if `guestok` is not defined or is set to `false`, guest access is disabled. By default, the guest access is disabled.

This procedure shows how to use the `zfs` command to enable guest access, but you can also use the `share` command for other file system types. See the `share`(1M) man page.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Enable guest access for a specified share.**

```
# zfs set share=name=share-name,path=/mntpnt/ds,prot=smb,guestok=true pool/dataset
```

Example 3–6 Setting the `guestok` Property to Enable Guest Access to an SMB Share

The following example uses the `zfs` command to enable guest access for the `myshare` share:

```
# zfs set share=name=myshare,path=/mntpnt/dir,prot=smb,guestok=true tank/home
name=myshare,path=/mntpnt/dir,prot=smb,guestok=true
```

If you attempt a connection to an SMB server without an account name or a valid account, the request is interpreted as a guest connection. Such a connection is not authenticated unless the guest account has a password. Windows systems typically use a predefined local account called `Guest` to represent guest connections. Note that this account can be renamed. In the Oracle Solaris OS, you can define an `idmap` name-based rule to map the `Guest` Windows user to any local Oracle Solaris user name, such as `guest` or `nobody`.

The following command creates a name-based mapping between the Windows user, `Guest`, and the Oracle Solaris user, `guest`:

```
# idmap add winname:Guest unixuser:guest
```

If the local account has an SMB password in the `/var/smb/smbpasswd` file, the guest connection is authenticated against that password. Any connection over SMB that is made by using an account that maps to the local guest account is designated as a guest connection. In the absence of an `idmap` rule for `Guest`, an ephemeral ID is generated for this Windows account by the `idmap` service.

▼ How to Enable Access-Based Enumeration for a Share

The access-based enumeration (ABE) feature filters directory content based on the access granted to the user who is browsing the directory. This feature is compatible with the Windows ABE feature.

When ABE filtering is enabled, you see *only* the files and directories to which you have access. This behavior has benefits such as the following:

- It is easier to find files in directories that contain many files by reducing the number of files shown in the listing.
- An "out-of-sight, out-of-mind" policy is implemented.

ABE filtering is managed on a per-share basis by using the `zfs` command to set the Boolean `abe` property. See the `zfs`(1M) man page.

ABE filtering is also supported on autohome shares. See the smbautohome(4) man page.

Note – With ABE filtering enabled, you still might see files in a directory listing that you cannot open. For example, if you have the ability to read the attributes of a file, ABE filtering shows the file in the directory listing, but you will be denied access if you attempt to open the file for reading or writing. Also, user privileges might result in files being shown, even though the ACL appears to deny all access.

When `abe=true`, ABE filtering is enabled on the share. Any directory entries to which you have no access are omitted from directory listings. When `abe=false` or is not defined, ABE filtering is not performed on the share. By default, the `abe` property is not defined.

This procedure shows how to use the `zfs` command to enable ABE filtering for a share, but you can also use the `share` command for other file system types. See the share(1M) man page.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Enable ABE filtering for a specified share.**

```
# zfs set share=name=share-name,path=/mntpnt/dir,prot=smb,abe=true pool/dataset
```

For example, the following command enables ABE filtering for the new `myshare` share:

```
# zfs set share=name=myshare,path=/mntpnt/dir,prot=smb,abe=true tank/home
name=myshare,path=/mntpnt/dir,prot=smb,abe=true
```

How to Modify SMB Share Properties (`zfs`)

Use this procedure to change properties on a share.

This procedure shows how to use the `zfs` command to modify share properties, but you can also use the `share` command for other file system types. See the share(1M) man page.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **View the existing share.**

```
# zfs get share tank/home
NAME      PROPERTY VALUE SOURCE
tank/home share    name=home,path=/tank/home,prot=smb,guestok=true,
csc=auto tank/home
```

3. **Modify the SMB share properties.**

For example, first change the `guestok` property to `false`.

```
# zfs set share=name=home,guestok=false tank/home
name=home,path=/tank/home,prot=smb,guestok=false,csc=auto
```

Then, change the value of the `csc` property from `auto` to `disabled`.

```
# zfs set share=name=home,prot=smb,csc=disabled tank/home
name=home,path=/tank/home,desc=HOME,prot=smb,guestok=true,csc=disabled
```

▼ How to Remove an SMB Share (`zfs`)

This procedure describes how to remove an SMB share. When you remove an SMB share, the definition of the share is removed from the server. You can re-create the share with the `zfs` command.

This procedure shows how to use the `zfs` command to remove a share, but you can also use the `unshare` command for other file system types. See the `unshare`(1M) man page.

1. **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2. **Remove an SMB share.**

# zfs set -c share=name=*share-name pool/dataset*

For example, the following command removes the `sales_share1` share from the `tank/sales` dataset:

```
# zfs set -c share=name=share_sales1 tank/sales
share 'share_sales1' was removed
```

▼ 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 22 and the `smbautohome`(4) man page.

1. **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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 22.

How to Restrict Client Host Access to an SMB Share (`zfs`)

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

For more information about the access control mechanisms that are used for shares, see "Access Control to Shares" on page 21.

This procedure shows how to use the `zfs` command to restrict client host access, but you can also use the `share` command for other file system types. See the `share`(1M) man page.

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 information about access lists, see the `share_smb`(1M) man page.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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

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

```
# zfs set share=name=name,path=pathname,prot=smb,ro=hostname[:hostname] pool/dataset
# zfs set share=name=name,path=pathname,prot=smb,rw=hostname[:hostname] pool/dataset
# zfs set share=name=name,path=pathname,prot=smb,none=hostname[:hostname] pool/dataset
```

hostname A host name, a netgroup, or an IP address

pool/dataset Name of the dataset being shared

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 `files/acme.sales.logs` share. The `mercury` and `venus` hosts have read-write access, `mars` has read-only access, and `neptune` has no access.

```
# zfs set share=name=acme_sales_logs,path=/files/acme.sales.logs,prot=smb,\
rw=mercury:venus,ro=mars,none=neptune files/acme.sales.logs
```

Managing SMB Groups (Task Map)

This section describes how to manage SMB groups and privileges for the SMB server.

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

For information about SMB groups and local users, see "Local SMB Groups" on page 24.

The following table points to the tasks that you can use to manage SMB groups through the SMB server.

Task	Description	For Instructions
Create an SMB group.	Create an SMB group to manage users.	"How to Create an SMB Group" on page 77
Add a member to an SMB group.	Add a member to an SMB group by using the `smbadm` command.	"How to Add a Member to an SMB Group" on page 78

Task	Description	For Instructions
Remove a member from an SMB group.	Remove a member from an SMB group by using the smbadm command.	"How to Remove a Member From an SMB Group" on page 78
Modify SMB group properties.	An SMB 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 SMB group if you modify the value of the description property.	"How to Modify SMB Group Properties" on page 79

You use the smbadm(1M) command to manage SMB groups on the system that runs the SMB server.

▼ How to Create an SMB Group

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

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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 SMB group.**

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

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

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

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

▼ How to Add a Member to an SMB Group

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Add a user to the SMB 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 (\) is a shell special character and must be quoted. For instance, escape the backslash with another backslash: *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 an SMB group, specify the Oracle 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 an SMB Group

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Remove a user from the SMB 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 (\) is a shell special character and must be quoted. For instance, escape the backslash with another backslash: *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 an SMB group, specify the Oracle 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 SMB Group Properties

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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

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

Configuring the WINS Service

This section provides information about configuring the SMB server as a client to the WINS service. For information about configuring other applicable services, see "Configuring the SMB Server – Process Overview" on page 14.

▼ How to Configure WINS

If you are integrating an SMB server 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 SMB Service Troubleshooting wiki.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

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
```

Enabling CATIA V4/V5 Character Translations

The CATIA V4 product is only available for UNIX systems, but the CATIA V5 product is available for both UNIX and Windows systems. When creating files, the CATIA V4 product includes characters in file names that are invalid on Windows systems, which causes interoperability issues when files need to be shared between CATIA V4 on UNIX and CATIA V5 on Windows.

The following table lists the character translations that are available in order to support CATIA V4/V5 interoperability between UNIX and Windows clients. Note that this character translation is only required for interoperability between CATIA V4 on UNIX and CATIA V5 on Windows, and is disabled by default.

TABLE 3–1 CATIA Character Translation Table

CATIA V4 UNIX Character	CATIA V5 Windows Character	CATIA V5 Character Description
"	¨ (0x00a8)	Dieresis
*	¤ (0x00a4)	Currency sign
/	ø (0x00f8)	Latin small letter O with stroke
:	÷ (0x00f7)	Division sign
<	« (0x00ab)	Left-pointing double angle quotation mark
>	» (0x00bb)	Right-pointing double angle quotation mark
?	¿ (0x00bf)	Inverted question mark
\	ÿ (0x00ff)	Latin small letter Y with dieresis
|	¦ (0x00a6)	Broken bar

How to the Enable CATIA Interoperability Feature

You can use the `zfs` command to specify whether to perform CATIA translation on a per-share basis by setting the `catia` property to `true`. By default, the value is `false`, which means that CATIA translation is not performed.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Enable CATIA translation for a share.**

```
# zfs set share=name=share-name,path=pathname,prot=smb,catia=true pool/dataset
name=share-name,path=pathname,prot=smb,catia=true
```

The following example shows how to enable CATIA translation for the `files/acme.sales.logs` share:

```
# zfs set share=name=acme.logs,path=/files/acme.sales.logs,prot=smb,catia=true \
files/acme.sales.logs
name=acme.logs,path=/files/acme.sales.logs,prot=smb,catia=true files/acme.sales.logs
```

Configuring SMB Printing (Task Map)

SMB printing enables you to gain access to all of the Common UNIX Printing System (CUPS) printers. Each printer can be made accessible as SMB shares. The share names match the printer names, and the shared path is inherited from the `print$` share that you create.

By default, support for SMB printing is disabled.

The following table points to the tasks that you can use to configure SMB printing.

Task	Description	For Instructions
Add a printer to the system.	Use the CUPS Print Manager tool to administer remote print queues.	"Remote Server Configuration" in *Oracle Solaris Administration: Common Tasks*
Share a printer.	Use the CUPS Print Manager tool to share a printer.	"How to Unshare or Share a Printer" in *Oracle Solaris Administration: Common Tasks*
Enable the SMB print service.	Use the `sharectl` command to enable the SMB print service.	"How to Enable the SMB Print Service" on page 82

▼ How to Enable the SMB Print Service

This procedure shows how to enable support for SMB printing on your Oracle Solaris system. Part of this procedure includes the creation of a share called `print$`. The share path can point to any directory, which is used as the spool path for all SMB shared printers. This share must exist before you can print.

SMB printing is disabled by default, due to the `print_enable` property being set to `false`.

Note – You *cannot* map the `print$` share as a disk share. Attempts to do so might result in the Password prompt being issued but access being denied. Such a failure is reported in the system log.

After SMB printing is enabled, you can use the Windows Add Printer wizard to attach your Windows client to shared printers. The SMB shared printers are connected to the network and can be selected by name.

1 **Become an administrator, obtain the `solaris.smf.value.shares` and `solaris.smf.manage.shares` RBAC authorizations, or use the SMB Management RBAC profile.**

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

2 **Create a share called `print$`.**

```
# zfs set share=name=print$,path=pathname,prot=smb pool-name/fsname
```

3 **Set permissions on the directory to permit root access.**

```
# chmod A=user:root:full_set:allow pathname
```

4 **Publish the share.**

```
# zfs set sharesmb=on pool-name/fsname
```

5 **Enable the SMB print service.**

```
# sharectl set -p print_enable=true smb
```

6 **Verify that the SMB print service is enabled.**

```
# sharectl get -p print_enable smb
```

If the SMB print service is enabled, the `print_enable` property is set to `true`.

7 **(Optional) Refresh the SMB service if a CUPS printer is added after the SMB print service is first enabled.**

```
# svcadm refresh smb/server
```

Example 3–7 Enabling the SMB Print Service

This example assigns the `print$` share to an existing directory, `/tank/printspool`, and enables the SMB print service.

```
# zfs set share=name=print$,path=/tank/printspool,prot=smb tank/printspool
# chmod A=user:root:full_set:allow /tank/printspool
# zfs set sharesmb=on tank/printspool
# sharectl set -p print_enable=true smb
```

CHAPTER 4

SMB Client Administration (Tasks)

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

This chapter covers the following topics:

- "Managing SMB Mounts in Your Local Environment (Task Map)" on page 85
- "Managing SMB Mounts in the Global Environment (Task Map)" on page 91

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

Up-to-date troubleshooting information is available on the Oracle Solaris SMB Service wiki (`http://wiki.genunix.org/wiki/index.php/OpenSolaris_CIFS_Service`).

Managing SMB Mounts in Your Local Environment (Task Map)

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

Task	Description	For Instructions
Join your SMB client to an Active Directory (AD) domain.	You can use the `kclient` command to join your SMB client to an AD domain.	"How to Configure a Kerberos Client for an Active Directory Server" in *Oracle Solaris Administration: Security Services*
Find the shares that are available on an SMB server in your domain.	From a particular SMB server, view the shares that you can mount on a directory that you own.	"How to Find Available SMB Shares on a Known File Server" on page 86

Task	Description	For Instructions
Mount an SMB 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 an SMB Share on a Directory You Own" on page 87
View the list of SMB shares that are mounted on the system.	View the list of mounted SMB shares.	"How to View the List of Mounted SMB Shares" on page 88
Unmount an SMB share from a directory that you own.	When you no longer need access to an SMB share, you can unmount it.	"How to Unmount an SMB 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 an SMB 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 AD or Kerberos, but which synchronize passwords between Oracle Solaris clients and their SMB servers.	"How to Configure the PAM Module to Store an SMB Persistent Password" on page 90
Delete a persistent password.	If you no longer want to store a persistent password, delete it.	"How to Delete an SMB Persistent Password" on page 91

▼ How to Find Available SMB 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 SMB 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 SMB shares.

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

```
$ smbadm show-shares [-A | -u username] [-t] server
```

The `-A` option enables you to view shares anonymously. You are not prompted for a password. The `-u` *username* option indicates the user to authenticate on the specified SMB server. The `-t` option shows a heading for the output. If neither the `-A` nor the `-u` option is specified, the user that is running the command is authenticated on the SMB server.

3 **When prompted, enter the password for the user that you specified on the SMB 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 SMB shares.**

The `smbadm show-shares -t` output shows the name of the share and an optional text description of the share.

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

```
$ smbadm show-shares -t solarsystem
Enter password:
SHARE                DESCRIPTION
netlogon    Network Logon Service
ipc$        Service (Samba Server)
tmp        Temporary file space
public        Public Stuff
ethereal
root        Home Directories
6 shares (total=6, read=6)
```

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

```
$ smbadm show-shares -A solarsystem
```

How to Mount an SMB 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 -r network/smb/client
```

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

```
$ smbadm show-shares [-A | -u username] [-t] server
```

The -A option enables you to view shares anonymously. You are not prompted for a password. The -u *username* option indicates the user to authenticate on the specified SMB server. The -t option shows a heading for the output. If neither the -A nor the -u option is specified, the user that is running the command is authenticated on the SMB server.

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

4 **Create a mount point on which to mount the share.**

```
$ mkdir mount-point
```

For example, to create a mount point called /tmp/mnt, type:

```
$ mkdir /tmp/mnt
```

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

```
$ mount -F smbfs [-o user=username,domain=domain-name,...] //server/share mount-point
```

For example, to mount the tmp share from the solarsystem server on the /tmp/mnt mount point, type:

```
$ mount -F smbfs //solarsystem/tmp /tmp/mnt
```

▼ How to View the List of Mounted SMB Shares

This procedure shows how to list all of the SMB 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 SMB mounts.**

Use one of the following commands to list the mounted SMB shares:

- **Use the mount command.**

  ```
  $ mount -v | grep ’type smbfs’
  //solarsystem/tmp on /mnt type smbfs read/write/setuid/devices/dev=5080000
    on Tue Mar 29 11:40:18 2011
  //solarsystem/files on /files type smbfs read/write/setuid/devices/dev=4800000
    on Mon Mar 28 22:17:56 2011
  ```

  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
  //solarsystem/tmp      1871312   70864 1800448     4%    /mnt
  //solarsystem/files    8067749    8017 7979055     1%    /files
  ```

▼ How to Unmount an SMB 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 an SMB server:

   - **Use the `mount` command.**

     ```
     $ mount -v | grep 'type smbfs'
     //solarsystem/tmp on /mnt type smbfs read/write/setuid/devices/dev=5080000
       on Tue Mar 29 11:40:18 2011
     //solarsystem/files on /files type smbfs read/write/setuid/devices/dev=4800000
       on Mon Mar 28 22:17:56 2011
     ```

   - **Use the `df -k -F smbfs` command.**

     ```
     $ df -k -F smbfs
     //solarsystem/tmp      1871312   70864 1800448     4%    /mnt
     //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 an SMB Persistent Password

Interactions with an SMB 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 SMB server that you use to access shares.

The password you store persists until the `smbadm remove-key` command is run for the user.

- **Store the persistent password for the SMB server.**

  ```
  $ smbadm add-key [-u username]
  ```

  You can specify the user name as one of the following name types:

  - An *isolated name* can be a single label, such as `terry`, or a user principal name (UPN), such as `terry@example.com`.
  - A *composite name* includes the domain name, which can be a host name. A composite name uses one of these formats: *domain*`\`*username*, *domain*`/`*username*, or *username*`@`*domain*.

  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.

  ```
  $ smbadm add-key -u terry@solarsystem
  Password for SOLARSYSTEM/terry:
  ```

▼ How to Configure the PAM Module to Store an SMB Persistent Password

When installed, the `pam_smbfs_login.so.1` module enables you to store a persistent password as if you had run the `smbadm add-key` 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 AD or Kerberos, but which synchronize passwords between Oracle Solaris clients and their SMB servers.

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

- **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 `smbadm add-key` command.

Note – The PAM module implements a privilege to permit it to run as superuser to store your password.

How to Delete an SMB Persistent Password

Use this procedure to delete persistent passwords that are stored by the `smbadm add-key` command.

- **Delete one or more persistent passwords for the specified user by doing one of the following:**
  - **To delete a single persistent password that was created by the user running the `smbadm remove-key` command, type:**

    ```
    $ smbadm remove-key -u username
    ```

    For example, the following command removes the persistent password for `terry@solarsystem`:

    ```
    $ smbadm remove-key -u terry@solarsystem
    ```

  - **To delete all persistent passwords that were created by the user running the `smbadm remove-key` command, type:**

    ```
    $ smbadm remove-key
    ```

    For example, when user `dana` runs the command, he removes all of the persistent passwords that he created. After the passwords are deleted, the user is prompted for a password each time that he or she performs an SMB transaction.

Managing SMB Mounts in the Global Environment (Task Map)

The following table points to the tasks that superuser can perform to manage SMB 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 SMB Share" on page 92

Task	Description	For Instructions
Customize the SMB environment by setting SMB properties.	Use the `sharectl` command to set SMB properties.	"How to Customize the SMB Environment in Oracle Solaris" on page 93
View the SMB property values.	Use the `sharectl` command to view SMB property values.	"How to View the SMB Environment Property Values" on page 94
Add an SMB share to an automounter map.	Use this procedure if you want an SMB share to be automatically mounted at boot time.	"How to Add an Automounter Entry for an SMB Share" on page 94

▼ How to Mount a Multiuser SMB 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 an administrator.**

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

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 -r network/smb/client
```

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

```
# smbadm show-shares [-A | -u username] [-t] 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 [-o user=username,domain=domain-name,...] //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 -o uid=terry,gid=staff,fileperms=0644 //solarsystem/tmp /sales-tools
```

In this example, the mount options enable users other than root to access the share. User terry and users who are members of the staff group can access the share with mode 0644.

When you mount a share, you can set the uid and gid mount options to specify the user and group owner of the share.

The values specified by these mount options are used to do the following:

- Specify the user and group to be used for local access checks. These checks are only used to determine which local users are permitted through the mount point. All other access checks are handled by the server.
- Determine the UID and GID that appear in file listings when the mounted share does not support "per-file security." Such shares might be shared CD-ROMs or Windows FAT volumes. Most shares support "per-file security," so the UID and GID that are shown in directory listings are derived from the file security properties.

▼ How to Customize the SMB Environment in Oracle Solaris

You can customize the SMB environment by using the sharectl(1M) command.

1 **Become an administrator or use the SMBFS Management RBAC profile, which is part of the File System Management profile.**

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

2 **Determine which properties you want to set.**

For a description of the properties, see the smb(4) man page.

3 **Set a property value for the global SMB environment.**

```
# sharectl set [-h] [-p property=value] ... smb
```

For example, to specify that client signing is required, type:

```
# sharectl set -p client_signing_required=true smb
```

▼ How to View the SMB Environment Property Values

You can view the SMB environment property values by using the sharectl(1M) command.

- **Determine which properties you want to view.**

  For a description of the properties, see the smb(4) man page.

  - **To view the value for a specific property, type:**

    ```
    $ sharectl get [-p property] ... smb
    ```

    For example, to view the values for the client_signing_required property, type:

    ```
    $ sharectl get -p client_signing_required smb
    ```

  - **To view all of the property values, type:**

    ```
    $ sharectl get smb
    ```

▼ How to Add an Automounter Entry for an SMB Share

You can add an SMB 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 without the need to specify a password, you must store a persistent password to mount the share. See "How to Store an SMB Persistent Password" on page 89.

Caution – When a user mounts a remote SMB 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 an administrator.**

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

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,dirperms=0555 //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 that can be accessed anonymously, which does not require a password, specify the noprompt option:

  ```
  /PUBLIC -noprompt,fstype=smbfs,dirperms=0555 //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 SMB share is mounted, a user can use regular Oracle Solaris commands to access the files. Automounted shares are automatically unmounted after a period of inactivity.

APPENDIX A

SMB DTrace Provider

This appendix provides information about the SMB DTrace provider, which enables you to use stable probe names to write DTrace scripts for the SMB server. DTrace is a feature of the Oracle Solaris operating system (OS).

This appendix covers the following topics:

SMB DTrace Overview

The SMB DTrace provider enables you to use stable probe names to write DTrace scripts for the SMB server. For more information about the dynamic tracing capabilities of the Oracle Solaris OS, see *Solaris Dynamic Tracing Guide* and the `dtrace`(1M) man page.

The SMB server supports the following two probe types for each SMB request:

- The *operation*`-start` probe is called *before* the request is executed.
- The *operation*`-done` probe is called *after* the request has been executed.

SMB DTrace Probes

You can see the list of available SMB DTrace probes by running the `dtrace -P smb -l` command.

The following table shows the operation probes and the specific SMB write (`arg[2]`) argument that is used by the probe, if applicable. For more information, see "SMB DTrace Arguments" on page 102.

TABLE A–1 SMB DTrace Probes

Probe Name	args[2] Value
smb:::op-Read-start	smbReadArgs_t *
smb:::op-Read-done	smbReadArgs_t *
smb:::op-ReadRaw-start	smbReadArgs_t *
smb:::op-ReadRaw-done	smbReadArgs_t *
smb:::op-ReadX-start	smbReadArgs_t *
smb:::op-ReadX-done	smbReadArgs_t *
smb:::op-Write-start	smbWriteArgs_t *
smb:::op-Write-done	smbWriteArgs_t *
smb:::op-WriteAndClose-start	smbWriteArgs_t *
smb:::op-WriteAndClose-done	smbWriteArgs_t *
smb:::op-WriteAndUnlock-start	smbWriteArgs_t *
smb:::op-WriteAndUnlock-done	smbWriteArgs_t *
smb:::op-WriteRaw-start	smbWriteArgs_t *
smb:::op-WriteRaw-done	smbWriteArgs_t *
smb:::op-WriteX-start	smbWriteArgs_t *
smb:::op-WriteX-done	smbWriteArgs_t *
smb:::op-CheckDirectory-start	Not applicable
smb:::op-CheckDirectory-done	Not applicable
smb:::op-Close-start	Not applicable
smb:::op-Close-done	Not applicable
smb:::op-CloseAndTreeDisconnect-start	Not applicable
smb:::op-CloseAndTreeDisconnect-done	Not applicable
smb:::op-ClosePrintFile-start	Not applicable
smb:::op-ClosePrintFile-done	Not applicable
smb:::op-Create-start	Not applicable
smb:::op-Create-done	Not applicable
smb:::op-CreateDirectory-start	Not applicable

TABLE A–1 SMB DTrace Probes *(Continued)*

Probe Name	`args[2]` Value
`smb:::op-CreateDirectory-done`	Not applicable
`smb:::op-CreateNew-start`	Not applicable
`smb:::op-CreateNew-done`	Not applicable
`smb:::op-CreateTemporary-start`	Not applicable
`smb:::op-CreateTemporary-done`	Not applicable
`smb:::op-Delete-start`	Not applicable
`smb:::op-Delete-done`	Not applicable
`smb:::op-DeleteDirectory-start`	Not applicable
`smb:::op-DeleteDirectory-done`	Not applicable
`smb:::op-Echo-start`	Not applicable
`smb:::op-Echo-done`	Not applicable
`smb:::op-Find-start`	Not applicable
`smb:::op-Find-done`	Not applicable
`smb:::op-FindClose-start`	Not applicable
`smb:::op-FindClose-done`	Not applicable
`smb:::op-FindClose2-start`	Not applicable
`smb:::op-FindClose2-done`	Not applicable
`smb:::op-FindUnique-start`	Not applicable
`smb:::op-FindUnique-done`	Not applicable
`smb:::op-Flush-start`	Not applicable
`smb:::op-Flush-done`	Not applicable
`smb:::op-GetPrintQueue-start`	Not applicable
`smb:::op-GetPrintQueue-done`	Not applicable
`smb:::op-Ioctl-start`	Not applicable
`smb:::op-Ioctl-done`	Not applicable
`smb:::op-LockAndRead-start`	`smbReadArgs_t *`
`smb:::op-LockAndRead-done`	`smbReadArgs_t *`
`smb:::op-LockByteRange-start`	Not applicable

TABLE A–1 SMB DTrace Probes *(Continued)*

Probe Name	`args[2]` Value
`smb:::op-LockByteRange-done`	Not applicable
`smb:::op-LockingX-start`	Not applicable
`smb:::op-LockingX-done`	Not applicable
`smb:::op-LogoffX-start`	Not applicable
`smb:::op-LogoffX-done`	Not applicable
`smb:::op-Negotiate-start`	Not applicable
`smb:::op-Negotiate-done`	Not applicable
`smb:::op-NtCancel-start`	Not applicable
`smb:::op-NtCancel-done`	Not applicable
`smb:::op-NtCreateX-start`	Not applicable
`smb:::op-NtCreateX-done`	Not applicable
`smb:::op-NtTransact-start`	Not applicable
`smb:::op-NtTransact-done`	Not applicable
`smb:::op-NtTransactSecondary-start`	Not applicable
`smb:::op-NtTransactSecondary-done`	Not applicable
`smb:::op-NtRename-start`	Not applicable
`smb:::op-NtRename-done`	Not applicable
`smb:::op-Open-start`	Not applicable
`smb:::op-Open-done`	Not applicable
`smb:::op-OpenPrintFile-start`	Not applicable
`smb:::op-OpenPrintFile-done`	Not applicable
`smb:::op-WritePrintFile-start`	Not applicable
`smb:::op-WritePrintFile-done`	Not applicable
`smb:::op-OpenX-start`	Not applicable
`smb:::op-OpenX-done`	Not applicable
`smb:::op-ProcessExit-start`	Not applicable
`smb:::op-ProcessExit-done`	Not applicable
`smb:::op-QueryInformation-start`	Not applicable

TABLE A–1 SMB DTrace Probes *(Continued)*

Probe Name	`args[2]` Value
`smb:::op-QueryInformation-done`	Not applicable
`smb:::op-QueryInformation2-start`	Not applicable
`smb:::op-QueryInformation2-done`	Not applicable
`smb:::op-QueryInformationDisk-start`	Not applicable
`smb:::op-QueryInformationDisk-done`	Not applicable
`smb:::op-Rename-start`	Not applicable
`smb:::op-Rename-done`	Not applicable
`smb:::op-Search-start`	Not applicable
`smb:::op-Search-done`	Not applicable
`smb:::op-Seek-start`	Not applicable
`smb:::op-Seek-done`	Not applicable
`smb:::op-SessionSetupX-start`	Not applicable
`smb:::op-SessionSetupX-done`	Not applicable
`smb:::op-SetInformation-start`	Not applicable
`smb:::op-SetInformation-done`	Not applicable
`smb:::op-SetInformation2-start`	Not applicable
`smb:::op-SetInformation2-done`	Not applicable
`smb:::op-Transaction-start`	Not applicable
`smb:::op-Transaction-done`	Not applicable
`smb:::op-TransactionSecondary-start`	Not applicable
`smb:::op-TransactionSecondary-done`	Not applicable
`smb:::op-Transaction2-start`	Not applicable
`smb:::op-Transaction2-done`	Not applicable
`smb:::op-Transaction2Secondary-start`	Not applicable
`smb:::op-Transaction2Secondary-done`	Not applicable
`smb:::op-TreeConnect-start`	Not applicable
`smb:::op-TreeConnect-done`	Not applicable
`smb:::op-TreeConnectX-start`	Not applicable

TABLE A–1 SMB DTrace Probes *(Continued)*

Probe Name	args[2] Value
smb:::op-TreeConnectX-done	Not applicable
smb:::op-TreeDisconnect-start	Not applicable
smb:::op-TreeDisconnect-done	Not applicable
smb:::op-UnlockByteRange-start	Not applicable
smb:::op-UnlockByteRange-done	Not applicable

SMB DTrace Arguments

This section describes the arguments that you use for the various SMB DTrace probes.

All probes use the first and second arguments, which are shown in the following code fragment:

```
args[0]        conninfo_t *        socket connection information
args[1]        smbopinfo_t *       SMB operation properties

typedef struct conninfo {
    string ci_local;        /* local host address */
    string ci_remote;       /* remote host address */
    string ci_protocol;     /* protocol (ipv4, ipv6, etc) */
} conninfo_t;

typedef struct smbopinfo {
    cred_t   *soi_cred;   /* credentials for operation */
    string   soi_curpath; /* current file handle path (if any) */
    uint64_t soi_sid;     /* session id */
    uint32_t soi_pid;     /* process id */
    uint32_t soi_status;  /* status */
    uint16_t soi_tid;     /* tree id */
    uint16_t soi_uid;     /* user id */
    uint16_t soi_mid;     /* request id */
    uint16_t soi_flags2;  /* flags2 */
    uint8_t  soi_flags;   /* flags */
} smbopinfo_t;
```

Read operation probes also use the third argument, which is shown in the following code fragment:

```
args[2]    smbReadArgs_t *

typedef struct smbReadArgs {
    off_t    soa_offset;
    uint_t    soa_count;
} smbReadArgs_t;
```

Write operation probes also use the third argument, which is shown in the following code fragment:

```
args[2]    smbWriteArgs_t *

typedef struct smbWriteArgs {
    off_t    soa_offset;
    uint_t    soa_count;
} smbWriteArgs_t;
```

SMB DTrace Examples

The following example DTrace script shows how to trace all SMB requests:

```
#!/usr/sbin/dtrace -s

#pragma D option quiet

dtrace:::BEGIN
{
        printf(
            "%39s/%-17s %-31s %8s %-10s %5s %9s %5s %6s %4s\n",
            "CLIENT",
            "SESSION",
            "REQUEST",
            "TIME(us)",
            "STATUS",
            "MID",
            "PID",
            "TID",
            "FLAGS2",
            "FLAGS");
}

dtrace:::END
{
        printf(
            "%39s/%-17s %-31s %8s %-10s %5s %9s %5s %6s %4s\n",
            "CLIENT",
            "SESSION",
            "REQUEST",
            "TIME(us)",
            "STATUS",
            "MID",
            "PID",
            "TID",
            "FLAGS2",
            "FLAGS");
}

smb:::op-Read-start,
smb:::op-ReadRaw-start,
smb:::op-ReadX-start,
smb:::op-LockAndRead-start,
smb:::op-Write-start,
smb:::op-WriteAndClose-start,
smb:::op-WriteAndUnlock-start,
smb:::op-WriteRaw-start,
smb:::op-WriteX-start,
smb:::op-CheckDirectory-start,
smb:::op-Close-start,
smb:::op-CloseAndTreeDisconnect-start,
smb:::op-ClosePrintFile-start,
smb:::op-Create-start,
smb:::op-CreateDirectory-start,
smb:::op-CreateNew-start,
smb:::op-CreateTemporary-start,
smb:::op-Delete-start,
smb:::op-DeleteDirectory-start,
smb:::op-Echo-start,
smb:::op-Find-start,
smb:::op-FindClose-start,
smb:::op-FindClose2-start,
smb:::op-FindUnique-start,
smb:::op-Flush-start,
smb:::op-GetPrintQueue-start,
smb:::op-Ioctl-start,
smb:::op-LockByteRange-start,
smb:::op-LockingX-start,
smb:::op-LogoffX-start,
smb:::op-Negotiate-start,
smb:::op-NtCancel-start,
smb:::op-NtCreateX-start,
smb:::op-NtTransact-start,
smb:::op-NtTransactSecondary-start,
smb:::op-NtRename-start,
smb:::op-Open-start,
smb:::op-OpenPrintFile-start,
smb:::op-WritePrintFile-start,
smb:::op-OpenX-start,
smb:::op-ProcessExit-start,
smb:::op-QueryInformation-start,
smb:::op-QueryInformation2-start,
smb:::op-QueryInformationDisk-start,
smb:::op-Rename-start,
smb:::op-Search-start,
smb:::op-Seek-start,
smb:::op-SessionSetupX-start,
smb:::op-SetInformation-start,
smb:::op-SetInformation2-start,
smb:::op-Transaction-start,
smb:::op-Transaction2-start,
smb:::op-Transaction2Secondary-start,
smb:::op-TransactionSecondary-start,
smb:::op-TreeConnect-start,
smb:::op-TreeConnectX-start,
smb:::op-TreeDisconnect-start,
smb:::op-UnlockByteRange-start
{
        self->thread = curthread;
        self->start = timestamp;
}

smb:::op-Read-done,
smb:::op-ReadRaw-done,
smb:::op-ReadX-done,
smb:::op-LockAndRead-done,
smb:::op-Write-done,
smb:::op-WriteAndClose-done,
smb:::op-WriteAndUnlock-done,
smb:::op-WriteRaw-done,
smb:::op-WriteX-done,
smb:::op-CheckDirectory-done,
smb:::op-Close-done,
smb:::op-CloseAndTreeDisconnect-done,
smb:::op-ClosePrintFile-done,
smb:::op-Create-done,
smb:::op-CreateDirectory-done,
smb:::op-CreateNew-done,
smb:::op-CreateTemporary-done,
smb:::op-Delete-done,
smb:::op-DeleteDirectory-done,
smb:::op-Echo-done,
smb:::op-Find-done,
smb:::op-FindClose-done,
smb:::op-FindClose2-done,
smb:::op-FindUnique-done,
smb:::op-Flush-done,
smb:::op-GetPrintQueue-done,
smb:::op-Ioctl-done,
smb:::op-LockByteRange-done,
smb:::op-LockingX-done,
smb:::op-LogoffX-done,
smb:::op-Negotiate-done,
smb:::op-NtCancel-done,
smb:::op-NtCreateX-done,
smb:::op-NtTransact-done,
smb:::op-NtTransactSecondary-done,
smb:::op-NtRename-done,
smb:::op-Open-done,
smb:::op-OpenPrintFile-done,
smb:::op-WritePrintFile-done,
smb:::op-OpenX-done,
smb:::op-ProcessExit-done,
smb:::op-QueryInformation-done,
smb:::op-QueryInformation2-done,
smb:::op-QueryInformationDisk-done,
smb:::op-Rename-done,
smb:::op-Search-done,
smb:::op-Seek-done,
smb:::op-SessionSetupX-done,
smb:::op-SetInformation-done,
smb:::op-Transaction-done,
smb:::op-SetInformation2-done,
smb:::op-Transaction2-done,
smb:::op-Transaction2Secondary-done,
smb:::op-TransactionSecondary-done,
smb:::op-TreeConnect-done,
smb:::op-TreeConnectX-done,
smb:::op-TreeDisconnect-done,
smb:::op-UnlockByteRange-done
/self->thread == curthread/
{
        printf("%39s/%-17d %-31s %8d 0x%08x %5d %9d %5d 0x%04x 0x%02x\n",
               args[0]->ci_remote,
               args[1]->soi_sid,
	            probename,
	            (timestamp - self->start) / 1000,
	            args[1]->soi_status,
	            args[1]->soi_mid,
	            args[1]->soi_pid,
	            args[1]->soi_tid,
	            args[1]->soi_flags2,
	            args[1]->soi_flags);
}
```

The following example DTrace script traces reads and writes, which shows how the third argument is passed to `read` and `write` probes:

```
#!/usr/sbin/dtrace -s

#pragma D option quiet

dtrace:::BEGIN
{
        printf(
            "%39s/%-17s %-31s %8s %-10s %-17s %-10s %s\n",
            "CLIENT",
            "SESSION",
            "REQUEST",
            "TIME(us)",
            "STATUS",
            "OFFSET",
            "COUNT",
            "FILE");
}

dtrace:::END
{
        printf(
            "%39s/%-17s %-31s %8s %-10s %-17s %-10s %s\n",
            "CLIENT",
            "SESSION",
            "REQUEST",
            "TIME(us)",
            "STATUS",
            "OFFSET",
            "COUNT",
            "FILE");
}

smb:::op-Read-start,
smb:::op-ReadRaw-start,
smb:::op-ReadX-start,
smb:::op-LockAndRead-start
{
        self->thread = curthread;
        self->start = timestamp;
}

/*
 * The following action is executed if the field ’soi_curpath’ is undefined (or
 * NULL).
 */
```

```
smb:::op-Read-done,
smb:::op-ReadRaw-done,
smb:::op-ReadX-done,
smb:::op-LockAndRead-done
/self->thread == curthread && args[1]->soi_curpath == NULL/
{
        printf("%39s/%-17d %-31s %8d 0x%08x 0x%016x 0x%08x %s\n",
               args[0]->ci_remote,
               args[1]->soi_sid,
               probename,
               (timestamp - self->start) / 1000,
               args[1]->soi_status,
               args[2]->soa_offset,
               args[2]->soa_count,
               "NULL");
}

/*
 * The following action is executed if the field 'soi_curpath' is defined (or
 * points to an actual file path).
 */
smb:::op-Read-done,
smb:::op-ReadRaw-done,
smb:::op-ReadX-done,
smb:::op-LockAndRead-done
/self->thread == curthread && args[1]->soi_curpath != NULL/
{
        printf("%39s/%-17d %-31s %8d 0x%08x 0x%016x 0x%08x %s\n",
               args[0]->ci_remote,
               args[1]->soi_sid,
               probename,
               (timestamp - self->start) / 1000,
               args[1]->soi_status,
               args[2]->soa_offset,
               args[2]->soa_count,
               args[1]->soi_curpath);
}

smb:::op-Write-start,
smb:::op-WriteAndClose-start,
smb:::op-WriteAndUnlock-start,
smb:::op-WriteRaw-start,
smb:::op-WriteX-start
{
        self->thread = curthread;
        self->start = timestamp;
}

/*
 * The following action is executed if the field 'soi_curpath' is undefined (or
 * NULL).
 */
smb:::op-Write-done,
smb:::op-WriteAndClose-done,
smb:::op-WriteAndUnlock-done,
smb:::op-WriteRaw-done,
smb:::op-WriteX-done
/self->thread == curthread && args[1]->soi_curpath == NULL/
{
        printf("%39s/%-17d %-31s %8d 0x%08x 0x%016x 0x%08x %s\n",
               args[0]->ci_remote,
               args[1]->soi_sid,
               probename,
               (timestamp - self->start) / 1000,
               args[1]->soi_status,
               args[2]->soa_offset,
               args[2]->soa_count,
               "NULL");
}

/*
 * The following action is executed if the field 'soi_curpath' is defined (or
 * points to an actual file path).
 */
smb:::op-Write-done,
smb:::op-WriteAndClose-done,
smb:::op-WriteAndUnlock-done,
smb:::op-WriteRaw-done,
smb:::op-WriteX-done
/self->thread == curthread && args[1]->soi_curpath != NULL/
{
        printf("%39s/%-17d %-31s %8d 0x%08x 0x%016x 0x%08x %s\n",
               args[0]->ci_remote,
               args[1]->soi_sid,
               probename,
               (timestamp - self->start) / 1000,
               args[1]->soi_status,
               args[2]->soa_offset,
               args[2]->soa_count,
               args[1]->soi_curpath);
}
```

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.

Common Internet File System (SMB) 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 an Oracle Solaris user and between an Oracle 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 an Oracle Solaris group.

identity mapping A process that enables Windows clients to transparently access SMB shares and remote services from the Oracle Solaris SMB server.

Lightweight Directory 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 Oracle 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 an SMB client to mount SMB shares without having to authenticate each mount action. This password remains in storage until removed by the `smbadm remove-key` command.

relative identifier (RID) A 32-bit identifier similar to an Oracle 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 Oracle Solaris users and groups by name rather than by identifier.

Samba An open source service that enables UNIX servers to provide SMB file-sharing and printing services to SMB 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 an Oracle Solaris SMB 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.

SMB client Software that enables a system to access SMB shares from a SMB server.

SMB server Software that enables a system to make SMB shares available to SMB clients.

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 an Oracle 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