Managing SMB File Sharing and Windows Interoperability in Oracle Solaris 11.4

E61013-01
November 2020

ORACLE

Managing SMB File Sharing and Windows Interoperability in Oracle Solaris 11.4,

E61013-01

Primary Author: Cathleen Reiher, Alta Elstad

Contents

2 Setting Up Identity Mapping Between Windows and Oracle Solaris Systems

3 Setting Up an Oracle Solaris SMB Server to Manage and Share Files

Using This Documentation

- **Overview** – Describes how to share files between an Oracle Solaris system and a Windows system by using an SMB server
- **Audience** – System administrators
- **Required knowledge** – Basic and some advanced network administration skills

Product Documentation Library

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

Feedback

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

1

Sharing Files Between Windows and Oracle Solaris Systems

The Oracle Solaris operating system (Oracle Solaris OS) now provides Windows interoperability with the introduction of an integrated Server Message Block (SMB) server. The SMB server provides the ability to share files between a Windows and an Oracle Solaris system.

This document provides the information you need to integrate an Oracle Solaris SMB server into an existing Windows environment. It focuses on the information required to integrate an SMB server and how to use the SMB client. Windows topics are covered only when those topics affect the integration of an SMB server into the Windows environment.

This chapter covers the following topics:

- SMB File Sharing Environment
- Configuring the SMB Server – Process Overview
- Utilities and Files Associated With the SMB Server and Client
- Authentication, Directory, Naming, and Time Services
- SMB Shares
- Local SMB Groups
- SMB Share Execution Properties
- SMB Support for the Distributed File System
- SMB Auditing

> **Note:**
>
> The Oracle Solaris OS provides a *Server Message Block (SMB)* protocol server and client implementations. The SMB server implementation includes support for numerous SMB dialects including SMB 3.0, SMB 2.1, SMB 2.0, and SMB 1. The SMB client implementation includes support for SMB 1.

About SMB File Sharing

An Oracle Solaris server can now be an active participant in a Windows active directory domain and provide ubiquitous, cross-protocol file sharing through Server Message Block (SMB) and the Network File System (NFS) protocol to clients in their native dialect.

To integrate the Oracle Solaris OS server, you must configure the Oracle Solaris SMB server and then configure the identity mapping between Windows and Oracle Solaris OS systems. When the Oracle Solaris SMB server is integrated, Windows systems can access files on the Oracle Solaris OS server by using the SMB protocol. You can also use the Oracle Solaris SMB client to access files on a Windows or Oracle Solaris SMB server.

Native Oracle Solaris systems can serve files by means of SMB *shares* to SMB enabled clients, such as Windows and Mac OS systems. A share is a local directory on a server that is accessible to SMB clients on the network. Each share is identified by a name on the network. An SMB client sees each share separately, and does not see the server's directory path to the shared directory.

> **Note:**
>
> For a quick reference to commonly used commands for SMB file sharing, see Commonly Used SMB File Sharing Commands.

SMB File Sharing Environment

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, user authentication is delegated to a domain controller.

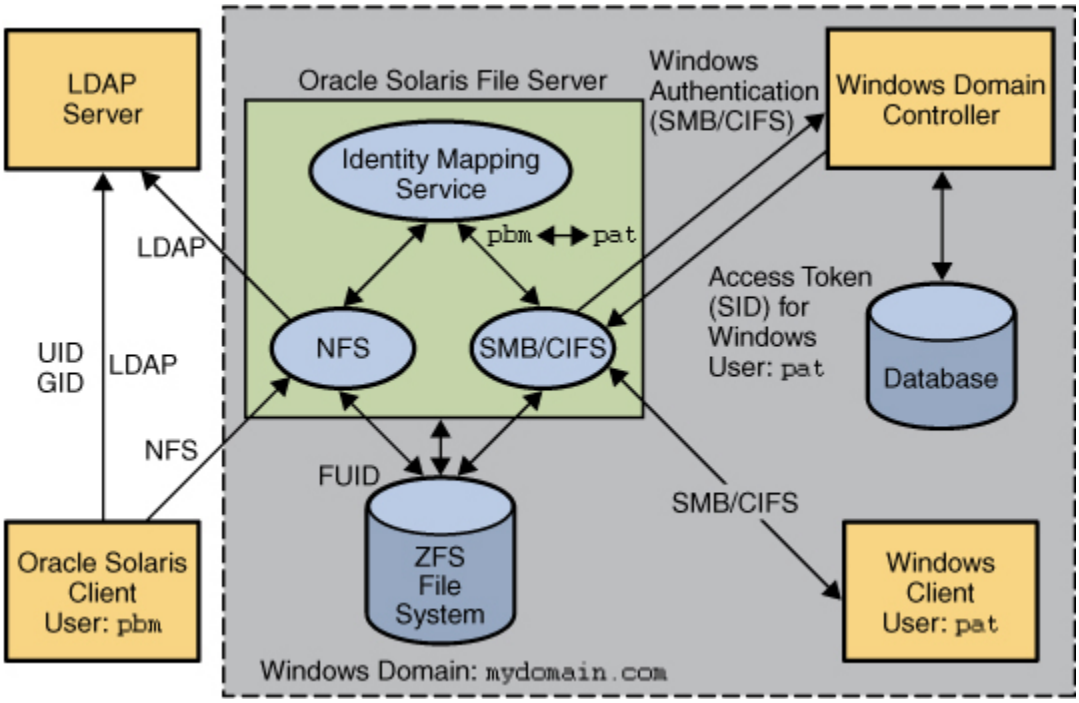
When a user requests access to a file or other resource, the server compares the user's identity and group memberships to the *access control list (ACL)* on the resource. Oracle Solaris and the ZFS file system have been enhanced to support Windows users and Windows-style access checking.

The Oracle Solaris OS is unique in that it can manage user identities simultaneously by using both traditional UIDs (and GIDs) and Windows identities. When a user logs in to the SMB server, the user's SMB identity is mapped to the appropriate UNIX® identity. This mapping is performed by using the `idmap` identity mapping service. If the Windows identity can be mapped to a UNIX identity, that identity is used. Otherwise, a temporary identity is generated by 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 a temporary ID needs to be stored on disk, the Windows identity is stored.

For more information about how the Oracle Solaris OS manages user identities, see Setting Up Identity Mapping Between Windows and Oracle Solaris Systems.

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

SMB Environment



The figure has the following components:

- **ZFS file system** – The ZFS file system is shared over the network by using the SMB and NFS protocols.
- **NFS** – The NFS server uses the NFS protocol to enable network clients to access the shared files.
- **SMB** – The SMB server uses the SMB protocol to enable network clients to access the shared files.
- **Windows client** – The Windows client accesses the shared resources over the network by using the SMB protocol.
- **Windows domain controller** – The Windows domain controller authenticates the Windows user `pat` when accessing shared resources on the SMB Server.
- **Identity mapping service** – The identity mapping service maps the Windows identities to Oracle Solaris UIDs and GIDs.
- **LDAP server** – The LDAP server uses LDAP to look up and authenticate NFS and Oracle Solaris users.

SMB Service Components

The Oracle Solaris SMB service includes the following components, described in this section:

- SMB Server
- SMB Client
- Identity Mapping Service

SMB Server

For a high-level overview of configuring the SMB server, see Configuring the SMB Server – Process Overview. For information about configuring the server, see Setting Up an Oracle Solaris SMB Server to Manage and Share Files. For more information about the SMB server, see the `smbadm`(8), `smbd`(8), `smbstat`(8), `smb`(5), `smbautohome`(5), and `pam_smb_passwd`(7) man pages.

The SMB features offered by the Oracle Solaris service depend on the file system that you are sharing. The ZFS file system fully supports Windows file sharing.

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 identity support, the file system must be able to store Windows identities.
- To provide full Windows ACL support, the file system must support NFS Version 4 ACLs.

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

For information about the supported features of the UFS file systems, see the `ufs`(4FS) man page. For information about the supported features of the ZFS file systems, see *Managing ZFS File Systems in Oracle Solaris 11.4*. For more information about NFS Version 4 ACLs, see ACLs and nfsmapid in NFS Version 4 in *Managing Network File Systems in Oracle Solaris 11.4*.

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

SMB Client

An Oracle Solaris user can use the SMB client to mount remote SMB shared directories. The SMB client enables an unprivileged user to mount and unmount shares on directories that the user owns. 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.

For more information about how to use the SMB client to access shares, see Using SMB File Sharing on Client Systems, and the `mount_smbfs`(8), `smbadm`(8), `smb`(5), `pam_smbfs_login`(7), and `smbfs`(4FS) 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 identities and Oracle Solaris user IDs and group IDs (UIDs and GIDs):

- **Directory-based mapping.**Uses mapping information that is stored in a name service directory along with other user or group information. The `idmap` service supports the following types of directory-based mappings:
  - **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 that is 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 `idmap` 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 Windows identity that is not already mapped. Ephemeral ID mapping is used by default.
- **Local ID mapping.** UNIX identities without explicit mappings are automatically mapped to equivalent Windows local identities.

You use `idmap` command to create, manage, and monitor mappings.

For more information about mapping user and group identities, see Mapping User and Group Identities. For information about how to determine your identity mapping strategy, see Creating Your Identity Mapping Strategy. For instructions about how to use the `idmap` command, see Managing Directory-Based Name Mapping for Users and Groups, Managing Rule-Based Identity Mapping for Users and Groups, and the `idmap`(8) 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`(8) and `smb`(5) man pages.

The `sharectl` command is used throughout the configuration process to set and view properties. This command and examples of its use are described in Setting Up an Oracle

Solaris SMB Server to Manage and Share Files and Using SMB File Sharing on Client Systems.

> **Note:**
>
> When the `smb/server_auth_level` value is `4`, the Oracle Solaris SMB server accepts both NTLM and NTLMv2 authentication mechanisms for local user authentication. When clients initiate NTLM authentication, the Oracle Solaris SMB server is required to use DES encryption algorithm for hashing the incoming password hashes. DES is known to be less secure than HMAC-MD5, which is used in generating NTLMv2 challenge responses.

> **Note:**
>
> The default value for `smb/server_auth_level` has changed from `4` to `5`. This change is made to limit the use of the less secure DES encryption mechanism for local user authentication. With the new default value, Oracle Solaris SMB server accepts only NTLMv2 authentication mechanisms for local user authentication. With this change, there is no impact to Windows clients running Windows Vista and later versions assuming the `LMCompatibilityLevel` registry setting is not down-level. However, Windows clients running a Windows OS version prior to Vista will require the `LMCompatibilityLevel` registry setting to be raised to `3` or higher in order to be successfully authenticated by the Oracle Solaris SMB server. For Domain users connecting to the Oracle Solaris SMB server, the AD domain controller will establish and enforce the required authentication level.

Configuring the SMB Server – Process Overview

This section describes the high-level process for configuring the SMB server. You might use the following services in your environment:

- **Domain Name System (DNS)** – For information about the DNS service, see *Working With Oracle Solaris 11.4 Directory and Naming Services: DNS and NIS*.
- **Kerberos** – For information about the Kerberos service, see *Managing Kerberos in Oracle Solaris 11.4*.
- **Lightweight Directory Access Protocol (LDAP)** – For information about the LDAP service, see Chapter 5, Setting Up LDAP Clients in *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.
- **Network Time Protocol (NTP)** – For information about the NTP service, see *Managing Clock Synchronization in Oracle Solaris 11.4*.
- **Windows Internet Naming Service (WINS)** – For information about the WINS service, see How to Configure WINS.

The procedures required to configure a SMB server are as follows:

1. Disable the Samba service, if necessary.

   See Disabling the Samba Service.

2. 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.
   - To join a workgroup, see How to Configure the SMB Server in Workgroup Mode.
3. Determine your identity mapping strategy.

   See Creating Your Identity Mapping Strategy.
4. Create one or more SMB shares.

   See How to Create an SMB Share (zfs).
5. Configure the Oracle Solaris system and the SMB Server as a client of the services that are used in your environment.

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:

- Managing SMB Servers and Clients by Using SMB Utilities (Task Map)
- SMB Service
- SMB Files

Managing SMB Servers and Clients by Using SMB Utilities (Task Map)

SMB utilities must be run as superuser or with specific privileges to be fully effective, but requests for some information can be made by all users. The following task map provides pointers to the tasks for managing SMB servers and clients.

Task	Description	For Instructions
Attach a remote SMB share to a specified mount point.	Use the `mount` command to mount an SMB share to a directory you own.	How to Mount an SMB Share on a Directory You Own How to Mount a Multiuser SMB Share How to Add an Automounter Entry for an SMB Share `mount_smbfs`(8) man page
Configure and manage file-sharing protocols such as SMB and NFS and network protocols such as NetBIOS.	Use the `sharectl` command to do the following: • Set client and server operational properties • Display property values for a specific protocol • Obtain the status of a protocol	How to Configure WINS Disabling and Re-enabling NetBIOS How to Customize the SMB Environment in Oracle Solaris How to View the SMB Environment Property Values `sharectl`(8) man page

Task	Description	For Instructions
Manage SMB shares on various file system types.	Use the `zfs` command to configure SMB sharing on Oracle Solaris ZFS file systems. Use the `share` command to manage SMB shares on various file system types.	How to Create an SMB Share (zfs) The `share`(8) and `zfs`(8) man pages For information about SMB share properties, see the `share_smb`(8) man page.
Manage domain membership of the SMB server.	The SMB server can use domain mode or workgroup mode. You can also use the `smbadm` command to configure SMB local groups.	How to Configure the SMB Server in Domain Mode How to Configure the SMB Server in Workgroup Mode How to Create an SMB Group How to Add a Member to an SMB Group How to Remove a Member From an SMB Group How to Modify SMB Group Properties `smbadm`(8) man page
Managing the Oracle Solaris SMB client.	Use the `smbadm` command to perform the following SMB client tasks: • View the shares available for mounting from a particular SMB server. • 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 server name to the NetBIOS workgroup and system name.	How to Find Available SMB Shares on a Known File Server How to Mount an SMB Share on a Directory You Own Storing SMB Persistent Passwords Configuring the PAM Module to Store an SMB Persistent Password Deleting an SMB Persistent Password How to Mount a Multiuser SMB Share
Display statistical information about the `smbd` server.	Use the `smbstat` command to show statistical information about the SMB server. By default, the `smbstat` command shows general information about the SMB server as well as dispatched SMB request counters.	`smbstat`(8) man page
Remove a named SMB share from a mount point.	Use the `umount` command to unmount a named SMB share.	How to Unmount an SMB Share From a Directory You Own `mount_smbfs`(8) man page

Task	Description	For Instructions
Create, modify, and remove SMB shares on ZFS file systems.	Use the `zfs set` command to set the SMB sharing property for the ZFS file system. Use the `zfs share` command to share a ZFS file system. Use the `zfs destroy` command to remove SMB shares from a ZFS file system.	How to Remove an SMB Share (zfs) `zfs`(8) man page
Create, modify, and remove SMB shares on non-ZFS file systems.	Use the `share` command to create an SMB share on non-ZFS file system types. Set the property values with the `share` command to modify an SMB share on non-ZFS file system types. Use the `unshare` command to remove SMB shares from non-ZFS file system types.	`unshare`(8) man page

SMB Service

The `svc:/network/smb/server` service provides the SMB service. The `svc:/network/smb/server` service depends on the following services:

- `svc:/network/smb/client:default`
- `svc:/system/idmap:default`

The `svc:/network/smb/client` service depends on the following services:

- `svc:/network/smb:default`
- `svc:/system/idmap:default`

> **Note:**
>
> The SMB service can be run only in the global zone.

SMB Files

The following files support SMB activities on Oracle Solaris systems:

- `/etc/auto_direct`
- `/etc/dfs/sharetab`
- `/etc/smbautohome`
- `/var/smb/smbpasswd`

`/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 Storing SMB Persistent Passwords.

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

`/etc/dfs/sharetab` File

The `/etc/dfs/sharetab` file contains a record of all the active shares in the system. Each entry in the file describes a share, which includes the mount point, share name, protocol, and share properties. See the `sharetab`(5) and `share_smb`(8) 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 SMB Autohome Shares and the `smbautohome`(5) man page.

`/var/smb/smbpasswd` File

The `/var/smb/smbpasswd` file stores the SMB passwords of Oracle Solaris users. The Oracle Solaris SMB server uses the SMB passwords to authenticate the connected Oracle Solaris users. For more information, see the `pam_smb_passwd`(7) 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 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.
- **DNS.** DNS resolves host names to Internet Protocol (IP) addresses for the system. This service enables you to identify a server by its name.
- **Dynamic DNS (DDNS).** DDNS is provided with AD and enables a client to dynamically update its entries in the DNS database.
- **UNIX Name Service.** You must be running a working UNIX name service, such as local files or LDAP, to share files.

- **NTP.** The NTP 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.
- **WINS.** A WINS server resolves NetBIOS names to IP addresses, which enables 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 function similar to a DNS server for NetBIOS names. For more information, see How to Configure WINS.

SMB Shares

A share makes a directory accessible to SMB clients on the network. Each share is identified by a name. An SMB client sees only the share name, not the server's path to the shared directory.

> **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 the user's autohome share will be listed.

SMB Share Properties

For information about creating an SMB share, see How to Create an SMB Share (zfs).

Use the `zfs set` and `zfs share` commands to set share properties that modify the attributes and behavior of an SMB share. For information about the `zfs set` and `zfs share` commands, see the `zfs`(8) man page.

For more information about setting share properties for ZFS file systems, see Sharing and Unsharing ZFS File Systems in *Managing ZFS File Systems in Oracle Solaris 11.4*.

For complete descriptions of the following properties, see the `share_smb`(8) and `zfs_share`(8) man pages. The two types of share properties are global and protocol-specific.

The global share properties include the following:

- `desc` – Specifies an optional description of the share
- `path` – Specifies the mount point of the share

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

- `abe` – Enables or disables access-based enumeration for a share

- `ad-container` – Specifies the name of an AD container in which to publish a share
- `catia` – Specifies whether to perform CATIA character substitution
- `cont_avail` – Enables or disables continuous availability to a share
- `csc` – Sets the client-side caching policy
- `dfsroot` – Enables or disables DFS root support on a share
- `encrypt` – Configures SMB encryption at the share level
- `guestok` – Enables or disables guest access to a share
- `none`, `ro`, `rw` – Sets host-based access rules for a share
- `oplocks` – Specifies the share-level oplocks configuration for the share
- `bypasstraverse` – Specifies whether to bypass traverse checking for the share

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.

Valid values for the `csc` property are:

- `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.

SMB Share Access Control

The SMB server uses the following access-control mechanisms to limit access to data shared by using SMB:

- **Host-based access control** limits access to shares based on which client system is making the request.
- **Share ACLs** limit user and group access to shares.
- **File and directory ACLs** limit user and group access to individual files and directories.

Host-based access control is applied first and grants or denies access to the client system. If the client system is granted access, the share ACL is then applied to grant or deny access to the user. Finally, the individual file and directory ACLs are consulted. You can access the data shared by using SMB only if all three access control mechanisms allow the access.

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 share ACLs to the share after the share is created.

Host-Based Access Control to SMB Shares

Host-based access control enables you to limit the access of a host or group of hosts to an SMB share. This host-based access control is enforced only for SMB access, not for local access or access through other protocols. 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). For more information about `share` command, see the `share`(8) man page. For more information about `zfs` command, see the `zfs`(8) man page. For more information about SMB shares, see the `share_smb`(8) man page. For information about the available options for sharing ZFS file system, see the `zfs_share`(8) man page.

Access Control Lists on SMB 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. For more information, 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. For more information, see Setting ACLs on ZFS Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

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. Access control lists are enforced only for SMB access, not for local access or access through other protocols. The default ACL setting permits full access to everyone.

Administrators with appropriate privileges can set ACL entries to audit access attempts to specific files. For more information about auditing events related to specific files, see New Feature – Per-Object Logging of Audit Events in *Managing Auditing in Oracle Solaris 11.4*. For information about how to specify files that must be audited, see Specifying Files or Directories to Be Audited in *Managing Auditing in Oracle Solaris 11.4*. For information about how to access SMB authentications using the audit tools, see SMB Auditing.

> **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.

SMB 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 `auser` and `buser`, you can manually define the shares as follows:

auser
`/home/auser`

buser
`/home/buser`

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/buser`, the autohome path is `/fort`. The temporary share is named `buser`. 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.

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:**
>
> If you are using autohome share, you cannot allow other users to access files in your home directory. All autohome shares are removed when the SMB server is restarted.

SMB 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
auser     /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 `auser` is `/home/a/auser`.

SMB Autohome Map Entry Format

A map entry uses the following format:

```
key
location [ container ]
```

key
Specifies a user name

location
Specifies the fully qualified path for the user's home directory

container
Specifies an optional AD container

An AD container name is specified as a comma-separated list of attribute name-value pairs. The attributes use the 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. For more information about AD container attribute names and values, see the `share_smb`(8) man page.

SMB 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 `auser`:

  ```
  auser /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/aa/auser`:

  ```
  auser /home/??/&
  ```

Wildcard Rule

When supplied in the key field, the asterisk (`*`) is recognized as the "catch-all" entry. This type of 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 applied *only* 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 asterisk wildcard entry, the `nsswitch` map is searched *only* 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

You can create local SMB groups on the system that runs the SMB server. These SMB groups apply only to users that are connected through SMB.

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.

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 share directories.

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 difference between an access right and a privilege is as follows:

- 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 cannot modify the privileges for the built-in SMB groups. However, you can assign any of the following privileges to the user-defined 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.

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 more information, see Managing SMB Groups and the `smbadm`(8) man page.

SMB 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 are 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 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 execute when the client connects to the share. The command can take the following arguments, which are substituted when the command is executed:
  - `%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 the host name is not available.
  - `%m` – Client NetBIOS name, or `""` if not available. This option is valid only 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 execute 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 has meaning only when the `map` property has been set. Otherwise, it has no effect.

  Valid values for the `disposition` property are:

  - `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 1-1 Using SMB 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.

SMB 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.

The SMB server supports only one standalone DFS namespace per system. Prior to creating a standalone DFS namespace from a Windows system, you must create an SMB share for the root of the namespace on the Oracle Solaris SMB server. When you configure the namespace, the SMB server sets the `dfsroot` property to enable the DFS root support on the SMB share. When you delete the namespace, the SMB server disables the DFS root support.

Use the DFS tools that are available on Windows systems to create and manage the standalone namespace in the Oracle Solaris OS.

SMB Auditing

All SMB authentications are recorded in Oracle Solaris audit logs and can be accessed by using the audit tools. The SMB authentications are logged for all domain or local users. The SMB events `AUE_smbd_session` and `AUE_smbd_logoff` are part of the `lo` class which audits logins, logouts, and screen locks. The `praudit` command enables you to get a list of audit trail records. The audit files are stored in the `/var/audit` directory.

To view the list of files in the `/var/audit` directory, use the following command:

```
$ ls -l /var/audit/
```

You can select a file in the `/var/audit` directory and view its records by using one of the following ways:

- To view all login and logout audit records in a file, use the following command:

  ```
  $ auditreduce -c lo /var/audit/filename | praudit -ls
  ```

- To view all SMB login audit records in a file, use the following command:

  ```
  $ auditreduce -c lo -m AUE_smbd_session /var/audit/filename | praudit -ls
  ```

- To view all SMB logout audit records in a file, use the following command:

  ```
  $ auditreduce -c lo -m AUE_smbd_logoff /var/audit/filename | praudit -ls
  ```

- To generate an HTML report of all login and logout audit records, use the following command:

  ```
  $ auditreduce -c lo /var/audit/* | praudit -x | xsltproc > filename.html
  ```

For more information about auditing in Oracle Solaris, see *Managing Auditing in Oracle Solaris 11.4*. You can also see the `auditreduce`(8) and `praudit`(8) man pages.

2

Setting Up Identity Mapping Between Windows and Oracle Solaris Systems

To successfully share files between your Oracle Solaris and Windows systems, you must establish an equivalence relationship between an Oracle Solaris user or group and a Windows user or group which ensures that they have equivalent rights on the system. The SMB server uses identity mapping to establish this equivalence relationship.

This chapter describes the `idmap` identity mapping service that maps Windows identities to Oracle Solaris user identities (UIDs) and group identities (GIDs). The chapter also includes instructions on how to manage name-based mappings. This chapter covers the following topics:

Mapping User and Group Identities

The SMB server resides in a multiprotocol environment and provides 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 has adopted the Windows model of access control lists (ACLs) by using ACLs in NFS Version 4 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.

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.

The `idmap` service supports the following types of mappings between Windows identifiers and Oracle Solaris user IDs and group IDs:

- **Directory-based mapping.** If configured, `idmap` first attempts to use mapping information that is stored in a directory with other user and group information.
  - **Directory-based name mapping.** In this mode, `idmap` attempts 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`(8) man page.
  - **Identity Management for UNIX (IDMU).** In this mode, `idmap` attempts 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 enables the administrator to define rules that associate Windows and Oracle Solaris users and groups by name.
- **Ephemeral ID mapping.** An *ephemeral ID* is a dynamic UID or GID mapping for a windows identity 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. 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 4, Setting Up Oracle Solaris Active Directory Clients in *Working With Oracle Solaris 11.4 Directory and Naming Services: DNS and NIS*.

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 vice versa. By default, rule-based mappings that you create are bidirectional.

The following example shows a bidirectional mapping of the Windows user `user3@example.com` to `uthree`, the Oracle Solaris user. Note that `user3@example.com` maps to `uthree`, and `uthree` maps to `user3@example.com`.

```
user3@example.com == uthree
```

For more information about other mapping types, see the `idmap`(8) 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 only 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. Use this method 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, 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 what one-to-one mappings offer, use mixed mode: both AD and native LDAP. 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.
- You can employ directory-based mapping *and* name-based rules.

Use the following method to configure directory-based mapping:

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

   > **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. Enable directory-based mapping, and inform the `idmap` service about the attributes to be used. See How to Configure Directory-Based Mapping.

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 you have to provide mappings for users and groups from other domains, you must use a different strategy, either in addition to or instead of IDMU. See How to Enable Identity Management for UNIX Support.

Using Rule-Based Mapping

This strategy 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 stored only on a particular system rather than being global. Use this method 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 `user4@example.com` to the Oracle Solaris user `user4`. The second command maps the Windows group called `staff@example.com` to the Oracle Solaris group `staff`.

   > **Note:**
   >
   > You can have only 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. Wildcard mappings for two domains would make determining which domain to use when mapping an Oracle Solaris user to a Windows user impossible.

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

   ```
   $ idmap add winuser:first@example.com unixuser:firstlast
   ```

   This command maps a Windows user called `first@example.com` to the Oracle Solaris user `firstlast`.

Mapping Well-Known Windows 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. This process 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 user3, which is not well known, and the well-known names administrator and guest:

```
$ idmap add winname:user3 unixuser:uthree
add     winname:user3    unixuser:uthree
$ 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:user3@example.com       unixuser:uthree
```

Managing Directory-Based Name Mapping for Users and Groups

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(8) command to manage identity mapping.

Task	Description	For Instructions
Extend the Active Directory (AD) schema with user and group name attributes.	Extends the AD schema and populates the user and group objects with UNIX user and group name information.	How to Extend the Active Directory Schema, and User and Group Entries
Extend the native LDAP schema with user and group name attributes.	Extends the native LDAP schema and populates the user and group objects with Windows user and group name information.	How to Extend the Native LDAP Schema, and User and Group Entries
Configure directory-based name mapping.	Enables 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

Task	Description	For Instructions
Add a directory-based name mapping to a user or group object.	Adds a directory-based name mapping to a user or group object in AD or native LDAP.	How to Add a Directory-Based Name Mapping to a User or Group Object
Remove a directory-based name mapping from a user or group object.	Removes a directory-based name mapping from a user or group object in AD or native LDAP.	How to Remove a Directory-Based Name Mapping From a User or Group Object

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

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

This procedure describes 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 a schema like the one in Extending the AD Schema, 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 Extending the AD Schema. 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.

   b. **Load the schema changes into AD from the Windows server.**

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

2. **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 or Group Object.

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 the sample LDIF file in Populating AD User and Group Objects. For more information about the LDIF file format, see RFC 2849.

b. **Obtain a Kerberos ticket-granting ticket (TGT) for a privileged AD principal.**

   The `ldapmodify` command uses this principal to update the AD objects described in the file you created.

   For example:

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

c. **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

This example shows a sample LDIF file, `ad_namemap_schema.ldif`, that 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
```

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

This example shows how to add Oracle Solaris user names to the appropriate user objects in AD by using the `ldapmodify` command. Windows users `user1`, `user2`, and `user3` are stored in Active Directory. These Windows users are associated with the Oracle Solaris users `uone`, `utwo`, and `uthree`, respectively.

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

```
$ cat updateUsers
dn: CN=User One,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: uone

dn: CN=User Two,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: utwo

dn: CN=User Three,CN=Users,DC=example,DC=com
changetype: modify
add: unixUserName
unixUserName: uthree
```

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 describes 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 a schema like the one in Extending the Native LDAP Schema, 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 Extending the Native LDAP Schema.

b. **Load the schema changes into native LDAP.**

```
$ ldapmodify -h LDAP-server-name -W -f schema-input.ldif
```

2. **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 or Group Object.

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

   See the sample LDIF file in Populating Native LDAP User and Group Objects. For more information about the LDIF file format, see RFC 2849.

b. **Update the user objects on the native LDAP server.**

```
$ ldapmodify -h LDAP-server-name -W -f user-input.ldif
```

Example 2-3 Extending the Native LDAP Schema

This example shows a sample LDIF file, `nldap_namemap_schema.ldif`, that 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 )
```

Load the schema changes into native LDAP. In the following example, the `-D` option argument changes a value that is specified in the `.ldif` file.

```
$ ldapmodify -h LDAP-server-name -D cn=admin -w - -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 `uone`, `utwo`, and `uthree` stored in native LDAP. These Oracle Solaris users are associated with the Windows users `user1`, `user2`, and `user3`, 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=uone,ou=passwd,dc=example,dc=com
changetype: modify
add: winAccountName
winAccountName: user1@example.com

dn: uid=utwo,ou=passwd,dc=example,dc=com
changetype: modify
add: winAccountName
winAccountName: user2@example.com

dn: uid=uthree,ou=passwd,dc=example,dc=com
changetype: modify
add: winAccountName
winAccountName: user3@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 and How to Extend the Native LDAP Schema, and User and Group Entries.

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 on the `idmap`(8) man page.

- `idmu` – Enables mapping by using Identity Management for UNIX (IDMU).

2. **Inform the idmap service about the new user and group attributes depending on the directory service or services you plan to use.**

> **Note:**
>
> Because these properties do not have default values, if they 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, issue the commands 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 for the UNIX user or group name to be stored in AD.

The following example 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 for the Windows name to be stored in native LDAP.

The following example 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 or Group Object

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

- Mapping a Windows user or group to an Oracle Solaris user or group by adding the Oracle Solaris user or group name to the AD object for the specified Windows user.
- Mapping an Oracle Solaris user or group to a Windows user by adding the Windows user or group name to the native LDAP object for the specified Oracle Solaris user or group.

For more information about the `idmap set-namemap` command and its options, see the `idmap`(8) man page.

1. **Become an administrator.**

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

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

   - Augment a user object.
     - To augment the Windows user object in AD:

       ```
       $ idmap set-namemap winuser:wusername@domain-name unixuser:uusername
       ```

       For example, the following command maps Windows user `first@example.com` to Oracle Solaris user `firstlast` by adding the Oracle Solaris name to the AD object for `first@example.com`.

       ```
       $ idmap set-namemap winuser:first@example.com unixuser:firstlast
       ```

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

       ```
       $ idmap set-namemap unixuser:uusername winuser:wusername@domain-name
       ```

       For example, the following command maps Oracle Solaris user `firstlast` to Windows user `first@example.com` by adding the Windows name to the native LDAP object for `firstlast`.

       ```
       $ idmap set-namemap unixuser:firstlast winuser:first@example.com
       ```

   - Augment a group object.
     - To augment the Windows group object in AD:

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

       For example, the following command maps the Windows group `salesgrp@example.com` to the 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:

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

       For example, the following command maps the Oracle Solaris group `sales` to the 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 or Group Object

1. **Become an administrator.**

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

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

   ```
   $ idmap get-namemap username
   ```

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

   - Remove the user name stored in the user object.
     - 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 `user3@example.com`.

       ```
       $ idmap unset-namemap winuser:user3@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 `uthree`.

       ```
       $ idmap unset-namemap unixuser:uthree
       ```

   - Remove the group name stored in the group object.
     - 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 the 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 the Oracle Solaris group `sales`.

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

Managing Directory-Based Identity Mapping by Using Identity Management for UNIX

This section describes how 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 Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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
   ```

About Rule-Based Identity Mapping for Users and Groups

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 have equivalent Windows identities 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.

Formatting Group and User Names

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.

You define the names of Oracle Solaris and Windows users and groups by using specific formats.

For Windows groups, use one of the following formats:

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

For Windows users, use one of the following formats:

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

For Oracle Solaris groups, use the format `unixgroup:`*group-name*.

For Oracle Solaris users, use the format `unixuser:`*username*.

> **Note:**
>
> Because only directional mappings can have an empty string (`""`) as their target identity, if you assign an empty string as a user name or group name, the identity service does not create a mapping and the `nobody` ID is used for access control. Therefore, to preclude logins by unmapped Windows users, do not assign an empty string as a user name or group name.

Using the wildcard character (`*`) matches all user names that are not matched by other mappings. Similarly, using the wildcard Windows name (`*@*`) matches all user names in all domains that are not matched by other mappings.

Using the wildcard on both sides of the mapping makes the user or group name the same for both Windows and Oracle Solaris users. For example, the `'*@example.com' == '*'` rule ensures that the `user1@example.com` Windows user name maps to the `user1` Oracle Solaris user name.

Note that the case of Windows names that appear in `idmap` name rules and in `idmap show` commands is ignored. However, because Windows names are not case sensitive but Oracle Solaris names are case sensitive, be careful when creating rule-based mappings that use wildcards for the user or group names.

> **Caution:**
>
> Using a wildcard to map Windows names to Oracle Solaris user names might not produce the expected results if user names contain uppercase characters. See the following example rules that handle this case.

Rule-based mapping rules that use the `unixuser:*` or `unixgroup:*` target map to the Oracle Solaris name as follows:

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

As a result of this differing treatment of case, names that appear to be alike might not be recognized as matches. You must create rules to handle such pairings, as shown in the following examples.

To map Oracle Solaris user `UserOne` to Windows user `userone@example.com`, you must create the following rule:

```
$ idmap add winuser:'*@example.com' unixuser:'*'
$ idmap add winuser:userone@example.com unixuser:UserOne
```

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
```

Managing Rule-Based Identity Mapping for Users and Groups

This section describes how to use the `idmap` command to manage identity mapping. 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`(8) command to manage identity mapping.

Task	Description	For Instructions
Add a user or group mapping rule.	Use `idmap` rules to create identity equivalents for Windows users or Windows groups and Oracle Solaris users or Oracle Solaris users based on the names in the naming services.	How to Add a User or Group Mapping Rule
Import rule-based user mappings from the `usermap.cfg` file.	Adds 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
List all of the mappings.	Review all mappings or to find particular mappings for users and groups.	Viewing Mapping Information
Show the mapping for a particular identity.	View how a particular name or ID is mapped.	Viewing a Mapping for a Particular Identity
Show all the established mappings.	View the mappings stored in the cache.	Viewing All Established Mappings
Remove a user or group mapping rule.	Removes a rule-based mapping when a user or group is no longer part of the naming service in your Windows domain.	How to Remove a User or Group Mapping Rule

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

Adding and Removing Group and User Mapping Rules

The `idmap` command to create rule-based mappings between Windows users and Oracle Solaris users.

How to Add a User or Group Mapping Rule

For information about formatting user and group names, see Formatting Group and User Names.

1. **Become an administrator.**

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

2. **Create the 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.

   - Create a user mapping.
     - To create a bidirectional mapping between a Windows user name and an Oracle Solaris user name:

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

     - To create a unidirectional mapping between a Windows user name and an Oracle Solaris user name:

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

     - To create a unidirectional mapping between an Oracle Solaris user name and a Windows user name:

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

   - Create a group mapping.

     If Windows uses a group identity as a file owner or a user identity as a file group, you need to create a diagonal mapping to map between a Windows group and an Oracle Solaris user and between an Oracle Solaris group and a Windows user.

     - To 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
       ```

     - To 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
       ```

     - To 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
       ```

     - To 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
       ```

- To 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 Remove a User or Group Mapping Rule

1. **Become an administrator.**

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

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

   ```
   $ idmap list
   ```

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

   ```
   $ idmap list | grep pat
   ```

   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 mappings.**

   - Remove one or more user mappings.
     - To remove any rule-based mapping that involves the specified user name, *username*:

       ```
       $ idmap remove username
       ```

     - To remove rule-based mappings between *username1* and *username2*:

       ```
       $ idmap remove username1 username2
       ```

     - To remove all rule-based mappings:

       ```
       $ idmap remove -a
       ```

   - Remove one or more group mappings.
     - To remove any rule-based mapping that involves the specified group name, *group-name*:

       ```
       $ idmap remove group-name
       ```

     - To remove rule-based mappings between *group-name1* and *group-name2*:

       ```
       $ idmap remove group-name1 group-name2
       ```

     - To remove all rule-based mappings:

       ```
       $ idmap remove -a
       ```

Importing User Mappings From a Rule-Mapping File

You use the `idmap import` command to import a set of rule-based user mappings that are stored in a file.

The `idmap` supports the following file formats:

- The NetApp `usermap.cfg` rule-mapping format:

  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:

  – == indicates a bidirectional mapping, which is the default

  – => or <= indicates a unidirectional mapping

  The IP qualifier is not supported.

- The Samba `smbusers` rule-mapping format:

  UNIX-name = *winname1 winname2* ...

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

  This format is based on the username map information on the `smb.conf` man page, which is available on the Samba 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 *UNIX-name*, if any. The following `idmap` command shows this mapping:

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

How to Import User Mappings From a Rule-Mapping File

1. **Become an administrator.**

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

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

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

   In the following example, a file called `myusermaps` uses the `usermap.cfg` format to specify the following user name mappings:

   ```
   $ cat myusermaps
   user3@example.com == user3
   userthree@example.com => user3
   ```

   You would 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
   ```

Viewing Mapping Information

This section describes how to show mapping information. The `idmap list` command 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.

- To list all of the mappings:

```
$ idmap list
add winuser:user1@example.com unixuser:uone
add wingroup:members unixgroup:staff
```

- To list only the user mappings:

```
$ idmap list | grep user
add winuser:user1@example.com unixuser:uone
```

- To list only the group mappings:

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

Viewing a Mapping for a Particular Identity

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

To 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, you would 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`, you would 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.

Viewing All Established Mappings

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

By default, the `idmap dump` command lists only 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:user3@a.user1.example.com <= uid:2147909633
winuser:user4@a.user1.example.com <= uid:2147909634
wingroup:Group Policy Creator Owners@a.user1.example.com == gid:2147917831
wingroup:Domain Admins@a.user1.example.com == gid:2147917832
wingroup:Enterprise Admins@a.user1.example.com == gid:2147917833
wingroup:Schema Admins@a.user1.example.com == gid:2147917834
wingroup:Netmon Users@a.user1.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.user1.example.com == uid:2147557377
usid:S-1-5-21-156362980-169493972-3399456007-513 == gid:2147557378
```

- To list only the user mappings:

  ```
  $ idmap dump -n | grep uid
  winuser:user3@a.user1.example.com <= uid:2147909633
  winuser:user4@a.user1.example.com <= uid:2147909634
  usid:S-1-5-21-156362980-169493972-3399456007-500 == uid:2147917825
  winuser:administrator@solar == uid:2147926017
  winuser:Administrator@a.user1.example.com == uid:2147557377
  ```

- To list only the group mappings:

  ```
  $ idmap dump -n | grep gid
  wingroup:Group Policy Creator Owners@a.user1.example.com == gid:2147917831
  wingroup:Domain Admins@a.user1.example.com == gid:2147917832
  wingroup:Enterprise Admins@a.user1.example.com == gid:2147917833
  wingroup:Schema Admins@a.user1.example.com == gid:2147917834
  wingroup:Netmon Users@a.user1.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
  ```

Troubleshooting the Identity Mapping Service

This section describes some troubleshooting tips for the identity mapping service. For related troubleshooting information, see the following sections:

- Troubleshooting the SMB Service
- Troubleshooting the SMB Client

Viewing Identity Mapping Service Property Settings

The identity mapping service uses the `svccfg` command to set properties. Before you change property values, you should view the current property settings.

To view configuration properties related to the `idmap` service, run the `svcprop -p config idmap` command. To show the debugging flags, run the `svccfg -s idmap listprop debug` command. For more information, see the `svccfg`(8) man page.

Saving and Restoring Name-Based Mapping Rules

You might need to back up and restore your name-based mapping rules.

How to Back Up and Restore Name-Based Mapping Rules

1. **List the name-based mapping rules and save them to a backup file.**

   ```
   $ idmap list >output-file
   ```

   For more information about the `idmap list` command, see the `idmap`(8) man page.

2. **Remove all the existing name-based mapping rules. If you attempt to restore the existing rules, duplicate entries might prevent you from restoring the backed up rules.**

   ```
   $ idmap remove -a
   ```

3. **Run the idmap command with the -f option to read and execute the idmap subcommands from the backup file.**

   ```
   $ idmap -f output-file
   ```

Viewing Details About Mappings

If you encounter unexpected mapping results, use the `-V` option of the `idmap dump` and `idmap show` commands to display detailed information about mappings.

For more information, see Viewing a Mapping for a Particular Identity.

Debugging the Identity Mapping Service

Through the `idmap` service, you can control the diagnostic verbosity in a number of areas. The `debug` property group defines several properties that control the debug verbosity in a particular area of the application. For all areas, the default is `0`, which produces error reports but no output in normal cases. The higher the value, the more verbosity is provided. Some properties support negative values to suppress reporting of errors.

The `debug/all` property acts as a master control. The effective value that is used for each area is the maximum of that area's property value and the value of `debug/all`. Thus, setting `debug/all` to a large value enables all available debugging output.

Output that is enabled is routed to `syslog` and the SMF service log, `/var/svc/log/system-idmap:default.log`. The `syslog.conf` settings further filter the logged information.

The following example shows how to use the `svccfg` command to set the property values and then use the `svcadm refresh` command to make them effective.

```
$ svccfg -s idmap setprop debug/discovery = 2
$ svcadm refresh idmap
$ svcprop -p debug idmap
```

The following table summarizes the initial debug output.

Property	Level	Output
`debug/config`	1	Configuration changes Loading configuration, beginning and end of discovery cycle Startup configuration Events that trigger reconfiguration Inability to discover domain configuration values
`debug/config`	2	Events that get noticed but do not trigger reconfiguration
`debug/mapping`	1	Mapping trace, as in `idmap show -V`
`debug/dns`	0	DNS errors
`debug/dns`	1	DNS queries and results
`debug/ldap`	0	LDAP authentication errors
`debug/ldap`	1	LDAP connection errors
`debug/discovery`	1	Result of AD domain service discovery step
`debug/discovery`	2	Starting discovery step Interim discovery results
`debug/door`	1	Report when request-processing threads are created or destroyed

3

Setting Up an Oracle Solaris SMB Server to Manage and Share Files

You can configure an Oracle Solaris SMB file server to run as a standalone server (workgroup mode) or to run in an existing Windows environment (domain mode). This type of server can provide access to SMB shares for both Oracle Solaris and Windows systems. This chapter describes how to configure an Oracle Solaris server, create and manage SMB shares, and customize the server and shares for your environment.

> **Note:**
>
> Currently, the SMB service runs only in the global zone.

This chapter covers the following topics:

- Disabling the Samba Service
- Configuring the SMB Server Operation Mode
- Managing SMB Shares
- Managing SMB Groups
- Configuring the WINS Service
- Enabling CATIA V4/V5 Character Translations
- Troubleshooting the SMB Service

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

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 Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

   ```
   $ svcs | grep samba
   ```

For example, the following output 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

This section describes how to configure the SMB server in domain mode or in workgroup mode.

The Active Directory (AD) service is a Windows 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.

Ensure that the naming service and the DNS service are configured correctly for the appropriate AD domain.

How to Configure the SMB Server in Domain Mode

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.

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 Join a Kerberos Client to an Active Directory Server in *Managing Kerberos in Oracle Solaris 11.4*.

Starting with the Oracle Solaris 11 OS, the `smbadm join` command automatically configures Kerberos.

1. **Become an administrator.**

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

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. **Ensure that the system clock on the Oracle Solaris system is within five minutes of the system clock of the domain controller (DC) by using following methods:**

   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
     ```

     If the NTP service is already running, then the `ntpdate` command fails with the following error:

     ```
     no server suitable for synchronization found.
     ```

     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 [-o organizational-unit] domain-name
   ```

   username
   Specifies an authenticated user account

   organizational-unit
   Specifies an alternative organizational unit in which to create a system's machine trust account

   domain-name
   Specifies a fully qualified NetBIOS or DNS domain name

   > **Note:**
   >
   > NetBIOS-based discovery is disabled if NetBIOS is disabled. See Disabling and Re-enabling NetBIOS.

   By default, a machine trust account for a system is automatically created in the default container for computer accounts (`cn=Computers`) as part of the domain join operation if the account does not already exist in Active Directory.

   For more information about the types of users who are permitted to perform a domain join operation and organizational units, see the `smbadm`(8) man page.

Example 3-1 Configuring the SMB Server in Domain Mode

The following examples show how to configure an SMB server in domain mode as a Domain Administrator and as an organizational unit (OU) administrator:

- The following example shows how a user with Domain Administrator privileges configures the SMB server in domain mode. User `domadmin` has Domain Administrator privileges. The name of the domain being joined is `westsales.example.com`.

```
$ svcadm enable -r smb/server
$ smbadm join -u domadmin 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'
```

- The following example shows how an OU administrator configures the SMB server in domain mode. An OU administrator does not have domain administrative privileges and can have control over one or more OUs. The name of the domain being joined is `westsales.example.com`.

  Based on the following hierarchy, a delegated administrator can create a machine trust account in one or more of the OUs:

```
dc=com
   dc=example
      dc=westsales
         ou=Departments
            ou=Engineering
            ou=Payables,Receivables,and Payroll
...
```

  The following examples show how designated administrators who do not have Domain Administrator privileges can configure an SMB server in a domain.

  – In the following example, user `deptadmin` is the designated administrator for the `Departments` OU. Because, `deptadmin` has already pre-staged the computer account in the `Departments` OU, the `-o` option is not required to add the server to the domain. The following command shows how `deptadmin` would run the `smbadm join` command:

```
$ smbadm join -u deptadmin westsales.example.com
```

  – In the following example, user `engadmin` is the designated administrator for the `Engineering` OU. The computer account has not been pre-staged, so `engadmin` must indicate the OU in which to create the account. The following command shows how `engadmin` creates the machine trust account in the `Engineering` OU:

```
$ smbadm join -u engadmin -o ou=Engineering,ou=Departments
westsales.example.com
```

  – In the following example, user `payadmin` is the designated administrator for the `Payables,Receivables,and Payroll` OU. The computer account has not been pre-staged, so `payadmin` must indicate the OU in which to create the account. The following command shows how `payadmin` creates the machine trust account in the `Payables,Receivables,and Payroll` OU:

```
$ smbadm join -u payadmin -o 'ou=Payables\,Receivables\,and
Payroll,ou=Departments' \
westsales.example.com
```

Note that the argument to the `-o` in the preceding command has escaped characters and is surrounded by single quotes (`'`). The following reserved characters must be escaped by using the backslash (`\`):

```
, + " \ < > ; = $
```

When you escape these reserved characters, you must also surround the string with single quotes because the backslash itself is a shell special character.

Additional Action

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).

How to Configure the SMB Server in Workgroup Mode

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.

In workgroup mode, the SMB server is responsible for authenticating users locally when access is requested to shared resources.

> **Note:**
>
> You can use the `smbadm` command to change from workgroup mode to domain mode.

1. **Become an administrator.**

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

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 workgroup other than the default WORKGROUP workgroup.**

   ```
   $ smbadm join -w workgroup-name
   ```

4. **Edit the `/etc/pam.d/other` 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:

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

   See the `pam_smb_passwd`(7) 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 this encrypted version of the password.

```
$ passwd username
```

Example 3-2 Configuring the SMB Server in Workgroup Mode

The following 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
```

Install the PAM module and generate the password for user `cal`.

```
$ echo "password required pam_smb_passwd.so.1 nowarn" >> /etc/pam.d/other
$ passwd cal
```

You would then create a share in order 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.

The Oracle Solaris OS introduced 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`, `share.smb`, and `share.nfs` properties. For information about Oracle Solaris command syntax, see Sharing and Unsharing ZFS File Systems in *Managing ZFS File Systems in Oracle Solaris 11.4*.

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 Sharing and Unsharing ZFS File Systems in *Managing ZFS File Systems in Oracle Solaris 11.4*.

Continuously Available Shares

SMB3 clients can request persistent file handles for a particular share by enabling the `cont_avail` property. Since there is an overhead involved in saving state of the share to the stable storage for persistent file handles, continuously available SMB shares are only recommended for enterprise applications that undergo a limited number of open and close operations. You must not share these SMB shares over NFS or on workloads such as the `/home` directory as these workloads have a high number of open and close operations. You can enable the `cont_avail` property on file systems only if the extended file attribute property, `xattr` is enabled.

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 by setting the `nbmand` option.	How to Enable Cross-Protocol Locking
Create an SMB share by using the ZFS file system's `share` property.	Makes a dataset available to clients.	How to Create an SMB Share (zfs)
Modify the properties of an SMB share by using the `share` command.	Changes share property values.	How to Modify SMB Share Properties (zfs)
Enable guest access to an SMB share.	Use the `zfs` command to enable guest access for a specified share by setting the `guestok` property.	How to Enable Guest Access to an SMB Share
Enable access-based enumeration (ABE) for an SMB share.	Use the `zfs` command to enable ABE filtering for a specified share by setting the `abe` property to `true`.	How to Enable Access-Based Enumeration for a Share
Remove an SMB share by using the `unshare` command.	When you remove a share, it can no longer be accessed by a system.	How to Remove an SMB Share (zfs)
Create an autohome share rule.	Specify custom share rules for autohome shares.	How to Create a Specific Autohome Share Rule
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)

About 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 set, the SMB server enforces mandatory share reservations and byte-range locking internally for all SMB clients. If the `nbmand` mount option is not set, there is limited coordination with NFS and local processes.

How to Enable Cross-Protocol Locking

1. **Become an administrator.**

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

2. **Set the `nbmand mount` option for an existing file system by using one of the following methods:**

   - 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 pool/dataset
     ```

     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.

   - Set the option by using the `zfs set` command.

     ```
     $ zfs set nbmand=on pool/dataset
     ```

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

     ```
     $ zfs set nbmand=on ztank/myfs
     ```

     > **Note:**
     >
     > The `nbmand` property takes effect only after the file system is unmounted and remounted.

Creating and Modifying SMB Shares

When you are using SMB, 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 `share.smb` property can be set to `on` or `off`. Specifying `share.smb=on` during dataset creation shares the dataset with the default share properties.

If the `zfs multilevel` property is set to `on`, then, only files and directories with the default label, `ADMIN_LOW`, are accessible to SMB clients.

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(8) 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.

1. **Become an administrator.**

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

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 vdev
   $ zfs create -o nbmand=on pool/dataset
   ```

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

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

3. **Enable SMB sharing for the ZFS file system on the dataset or on individual specified shares.**

   To enable SMB sharing on the dataset, set the share.smb property to on.

   ```
   $ zfs set share.smb=on pool/dataset
   ```

   To enable SMB sharing on individual named shares, first set share.smb=off on the dataset and then set share.smb=on on the individual shares.

   > **Note:**
   >
   > The zfs command automatically constructs the default share name which 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.**

   ```
   $ zfs share -o share.smb=on pool/dataset%share-name
   ```

   Use the zfs command to set share properties. See the zfs(8) man page.

   Share properties are stored as ZFS dataset properties, and the share ACL for each share is stored in the .zfs/shares directory of the dataset.

   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, create the dataset and share:

```
$ zfs create -o mountpoint=/users tank/users
$ zfs share -o share.smb=on tank/users%ushare
```

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

   For more information about SMB share properties, see SMB Share Properties, and the share_smb(8), share(8), and zfs(8) man pages.

6. **Verify how the file system is shared by using one of the following methods:**

   - Use the `zfs get` command.

     ```
     $ zfs get share.smb.all tank/admins%ashare
     NAME                PROPERTY                VALUE  SOURCE
     tank/admins%ashare  share.smb.abe           off    default
     tank/admins%ashare  share.smb.ad-container         default
     tank/admins%ashare  share.smb.catia         off    default
     tank/admins%ashare  share.smb.cont_avail    off    default
     tank/admins%ashare  share.smb.csc           auto   local
     tank/admins%ashare  share.smb.dfsroot       off    default
     tank/admins%ashare  share.smb.guestok       on     local
     tank/admins%ashare  share.smb.none                 default
     tank/admins%ashare  share.smb.ro                   default
     tank/admins%ashare  share.smb.rw                   default
     ```

   - Use the `share` command.

     ```
     $ share
     IPC$     smb     -       Remote IPC
     ashare  /admins  smb     csc=auto,guestok=true
     ```

   - View the `/etc/dfs/sharetab` file.

     ```
     $ cat /etc/dfs/sharetab
     -       IPC$    smb - Remote IPC
     /admins ashare smb guestok,csc=auto
     ```

Example 3-3 Creating a Share With the Client-Side Caching Policy Set to `auto`

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

```
$ zfs create -o mountpoint=/admins tank/admins
$ zfs share -o share.smb=on -o share.smb.csc=auto tank/admins%ashare
```

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

```
$ zfs set share.smb.guestok=on tank/admins%ashare
```

Example 3-4 Inherited SMB Sharing for ZFS File Systems in a Pool

For information about ZFS share property inheritance, see Sharing and Unsharing ZFS File Systems in *Managing ZFS File Systems in Oracle Solaris 11.4*.

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.

```
$ zfs create rpool/admins/user1
$ zfs create rpool/admins/user2
$ zfs set share.smb=on rpool/admins
$ zfs get -r share.smb rpool/admins
NAME                PROPERTY   VALUE   SOURCE
```

```
rpool/admins          share.smb  on     local
rpool/admins%         share.smb  on     inherited from rpool/admins
rpool/admins/user1    share.smb  on     inherited from rpool/admins
rpool/admins/user1%   share.smb  on     inherited from rpool/admins
rpool/admins/user2    share.smb  on     inherited from rpool/admins
rpool/admins/user2%   share.smb  on     inherited from rpool/admins
$ zfs set share.smb=off rpool/admins/user2
$ zfs get -r share.smb rpool/admins
NAME                  PROPERTY   VALUE  SOURCE
rpool/admins          share.smb  on     local
rpool/admins%         share.smb  on     inherited from rpool/admins
rpool/admins/user1    share.smb  on     inherited from rpool/admins
rpool/admins/user1%   share.smb  on     inherited from rpool/admins
rpool/admins/user2    share.smb  off    local
```

Example 3-5 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 system1 c0t3d0
$ zfs create -o share.smb=on -o nbmand=on system1/fs1
```

In this example, the share name `system1_fs1` is based on the dataset mount point `system1/fs1`.

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

```
$ zfs get -r share.smb system1/fs1
NAME          PROPERTY   VALUE  SOURCE
system1/fs1   share.smb  on     local
system1/fs1%  share.smb  on     inherited from system1/fs1
```

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

The `zfs get` command shows a subset of the share properties:

```
$ zfs get share.smb.all system1/fs1%
NAME          PROPERTY                VALUE  SOURCE
system1/fs1%  share.smb.abe           off    default
system1/fs1%  share.smb.ad-container         default
system1/fs1%  share.smb.catia         off    default
system1/fs1%  share.smb.cont_avail    off    default
system1/fs1%  share.smb.csc                  default
system1/fs1%  share.smb.dfsroot       off    default
system1/fs1%  share.smb.guestok       off    default
system1/fs1%  share.smb.none                 default
system1/fs1%  share.smb.ro                   default
system1/fs1%  share.smb.rw                   default
```

To view the local and inherited share properties, use the following command:

```
$ zfs get -rs local,inherited -e share.smb.all system1
NAME                 PROPERTY           VALUE      SOURCE
system1/fs1          share.smb.guestok  on         local
system1/fs1%         share.smb.guestok  on         inherited from system1/fs1
system1/fs2          share.smb.guestok  on         local
system1/fs2          share.smb.ro       otherhost  local
system1/fs2          share.smb.rw       myhost     local
system1/fs2%myshare  share.smb.guestok  on         inherited from system1/fs2
```

```
system1/fs2%myshare  share.smb.ro       otherhost  inherited from system1/fs2
system1/fs2%myshare  share.smb.rw       myhost     inherited from system1/fs2
```

To view all the share properties, use the following command:

```
$ zfs get share.all system1/fs1%
NAME          PROPERTY         VALUE         SOURCE
system1/fs1%  share.desc                     default
system1/fs1%  share.name       system1_fs1   -
system1/fs1%  share.nfs        off           default
system1/fs1%  share.nfs.*      ...           default
system1/fs1%  share.path                     default
system1/fs1%  share.point      /system1/fs1  -
system1/fs1%  share.protocols  smb           inherited from system1/fs1
system1/fs1%  share.smb        on            inherited from system1/fs1
system1/fs1%  share.smb.*      ...           default
system1/fs1%  share.state      shared        -
```

A property value of `...` can be expanded further by using the `.all` keyword. For example, you can view the `share.smb.*` properties by using the following command:

```
$ zfs get share.smb.all system1/fs1%
NAME          PROPERTY                VALUE         SOURCE
system1/fs1%  share.smb.abe           off           default
system1/fs1%  share.smb.ad-container                default
system1/fs1%  share.smb.catia         off           default
system1/fs1%  share.smb.cont_avail    off           default
system1/fs1%  share.smb.csc                         default
system1/fs1%  share.smb.dfsroot       off           default
system1/fs1%  share.smb.guestok       off           default
system1/fs1%  share.smb.none                        default
system1/fs1%  share.smb.ro                          default
system1/fs1%  share.smb.rw                          default
```

You can also view both the global share properties and the SMB properties by using the following command:

```
$ zfs get share.all,share.smb.all system1/fs1%
NAME          PROPERTY                VALUE         SOURCE
system1/fs1%  share.desc                            default
system1/fs1%  share.name              system1_fs1   -
system1/fs1%  share.nfs               off           default
system1/fs1%  share.nfs.*             ...           default
system1/fs1%  share.path                            default
system1/fs1%  share.point             /system1/fs1  -
system1/fs1%  share.protocols         smb           inherited from system1/fs1
system1/fs1%  share.smb               on            inherited from system1/fs1
system1/fs1%  share.smb.*             ...           default
system1/fs1%  share.state             shared        -
system1/fs1%  share.smb.abe           off           default
system1/fs1%  share.smb.ad-container                default
system1/fs1%  share.smb.catia         off           default
system1/fs1%  share.smb.cont_avail    off           default
system1/fs1%  share.smb.csc                         default
system1/fs1%  share.smb.dfsroot       off           default
system1/fs1%  share.smb.guestok       off           default
system1/fs1%  share.smb.none                        default
system1/fs1%  share.smb.ro                          default
system1/fs1%  share.smb.rw                          default
```

The following commands create another file system in the `system1` pool called `fs2`, associate the file system with the `myshare` share name, and enable SMB sharing:

```
$ zfs create -o nbmand=on system1/fs2
$ zfs share -o share.smb=on system1/fs2%myshare
```

You can use the `zfs get` command to view the `share.smb` and `share` property values for the `system1` pool.

```
$ zfs get -r share.smb.all system1
NAME                  PROPERTY   VALUE  SOURCE
system1               share.smb  off    default
system1/fs1           share.smb  on     local
system1/fs1%          share.smb  on     inherited from system1/fs1
system1/fs2           share.smb  off    default
system1/fs2%myshare   share.smb  on     local
```

```
$ zfs get -r share.smb.all system1
NAME                  PROPERTY                VALUE  SOURCE
system1               share.smb.abe           off    default
system1               share.smb.ad-container         default
system1               share.smb.catia         off    default
system1               share.smb.cont_avail    off    default
system1               share.smb.csc                  default
system1               share.smb.guestok       off    default
system1               share.smb.none                 default
system1               share.smb.ro                   default
system1               share.smb.rw                   default
system1/fs1           share.smb.abe           off    default
system1/fs1           share.smb.ad-container         default
system1/fs1           share.smb.catia         off    default
system1/fs1           share.smb.cont_avail    off    default
system1/fs1           share.smb.csc                  default
system1/fs1           share.smb.guestok       off    default
system1/fs1           share.smb.none                 default
system1/fs1           share.smb.ro                   default
system1/fs1           share.smb.rw                   default
system1/fs1%          share.smb.abe           off    default
system1/fs1%          share.smb.ad-container         default
system1/fs1%          share.smb.catia         off    default
system1/fs1%          share.smb.cont_avail    off    default
system1/fs1%          share.smb.csc                  default
system1/fs1%          share.smb.dfsroot       off    default
system1/fs1%          share.smb.guestok       off    default
system1/fs1%          share.smb.none                 default
system1/fs1%          share.smb.ro                   default
system1/fs1%          share.smb.rw                   default
system1/fs2           share.smb.abe           off    default
system1/fs2           share.smb.ad-container         default
system1/fs2           share.smb.catia         off    default
system1/fs2           share.smb.cont_avail    off    default
system1/fs2           share.smb.csc                  default
system1/fs2           share.smb.guestok       off    default
system1/fs2           share.smb.none                 default
system1/fs2           share.smb.ro                   default
system1/fs2           share.smb.rw                   default
system1/fs2%myshare   share.smb.abe           off    default
system1/fs2%myshare   share.smb.ad-container         default
system1/fs2%myshare   share.smb.catia         off    default
system1/fs2%myshare   share.smb.cont_avail    off    default
system1/fs2%myshare   share.smb.csc                  default
```

```
system1/fs2%myshare  share.smb.dfsroot       off    default
system1/fs2%myshare  share.smb.guestok       off    default
system1/fs2%myshare  share.smb.none                 default
system1/fs2%myshare  share.smb.ro                   default
system1/fs2%myshare  share.smb.rw                   default
```

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 `system1/fs2` called `system1/fs2/fs2_sub1`:

```
$ zfs create system1/fs2/fs2_sub1
```

The new file system inherits the `share.smb` property from its parent, `system1/fs1`, which causes a new default share to be created.

```
$ zfs create -o nbmand=on system1/fs1/fs1_sub1
$ zfs get -r share.smb system1
NAME                   PROPERTY   VALUE  SOURCE
system1                share.smb  off    default
system1/fs1            share.smb  on     local
system1/fs1%           share.smb  on     inherited from system1/fs1
system1/fs1/fs1_sub1   share.smb  on     inherited from system1/fs1
system1/fs1/fs1_sub1%  share.smb  on     inherited from system1/fs1
system1/fs2            share.smb  off    default
system1/fs2%myshare    share.smb  on     local
system1/fs2/fs2_sub1   share.smb  off    default
```

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

```
$ cat /etc/dfs/sharetab
/system1/fs2              myshare                  smb     -
/system1/fs1              system1_fs1              smb     -
/system1/fs1/fs1_sub1     system1_fs1_fs1_sub1     smb     -
```

If you disable SMB sharing for `system1/fs1`, that file system and its children are affected.

```
$ zfs set share.smb=off system1/fs1
$ zfs get -r share.smb system1
NAME                   PROPERTY   VALUE  SOURCE
system1                share.smb  off    default
system1/fs1            share.smb  off    local
system1/fs1/fs1_sub1   share.smb  off    inherited from system1/fs1
system1/fs2            share.smb  off    default
system1/fs2%myshare    share.smb  on     local
system1/fs2/fs2_sub1   share.smb  off    default

$ cat /etc/dfs/sharetab | grep system1
/system1/fs2      myshare smb     -
```

Note that disabling the `share.smb` property unpublishes the shares but does not remove the share definitions. The `/etc/dfs/sharetab` file shows that only the `myshare` share is still published, while the `system1_fs1` and `system1_fs2_fs2_sub1` shares still exist but are no longer published.

Example 3-6 Setting the csc Property for Shares

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

First, create and share a file system.

If you specify `share.smb=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 (_).

The automatic (auto) share is represented as `tank/zvol%`, which is the ZFS property name for the auto share. The default share name is constructed from the file system name. Invalid characters are converted to underscores. The `share.name` property stores the default share name, which is the name by which the share is published. The following example uses a default share name of `tank_zvol`.

```
$ zfs create -o utf8only=on -o share.smb=on tank/zvol
$ share
IPC$                            smb     -       Remote IPC
c$              /var/smb/cvol   smb     -       Default Share
tank_zvol       /tank/zvol      smb     -
$ zfs get name,share.protocols,share.state,share.point tank/zvol%

NAME        PROPERTY         VALUE       SOURCE
tank/zvol%  name             tank/zvol%  -
tank/zvol%  share.protocols  smb         local
tank/zvol%  share.state      shared      -
tank/zvol%  share.point      /tank/zvol  -
```

To list automatic shares, use the `zfs list -o share` command:

```
$ zfs create -o utf8only=on -o share.smb=on tank/zvol
$ zfs get share tank/zvol%
$ zfs list -o share
NAME        SHARENAME  PROTOCOLS  STATE        SHAREPOINT
tank/zvol%  tank_zvol  smb        shared       /tank/zvol
$ zfs get share.name tank/zvol%
NAME        PROPERTY    VALUE      SOURCE
tank/zvol%  share.name  tank_zvol  -
```

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

1. Create the dataset.

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

2. Create and enable an SMB share with the name of `ashare`.

   ```
   $ zfs share -o share.smb=on tank/zvol%ashare
   $ zfs get name,share.protocols,share.state,share.point tank/zvol%ashare
   NAME                   PROPERTY        VALUE                  SOURCE
   tank/zvol%ashare       name            tank/zvol%ashare       -
   tank/zvol%ashare       share.protocols smb                    local
   tank/zvol%ashare       share.state     -                      -
   tank/zvol%ashare       share.point     /tank/zvol             -
   ```

3. View the active shares on the system.

   ```
   $ cat /etc/dfs/sharetab
   /tank/zvol ashare smb -
   ```

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

```
$ zfs share -o share.smb=on -o share.smb.csc=auto tank/zvol%bshare
$ zfs get share.smb.all tank/zvol%bshare
NAME                  PROPERTY                VALUE                   SOURCE
tank/zvol%bshare      name                    tank/zvol%bshare        -
tank/zvol%bshare      share.smb.abe           off                     default
tank/zvol%bshare      share.smb.ad-container                          default
tank/zvol%bshare      share.smb.catia         off                     default
tank/zvol%bshare      share.smb.cont_avail    off                     default
tank/zvol%bshare      share.smb.csc           auto                    local
tank/zvol%bshare      share.smb.dfsroot       off                     default
tank/zvol%bshare      share.smb.guestok       off                     default
tank/zvol%bshare      share.smb.none                                  default
tank/zvol%bshare      share.smb.ro                                    default
tank/zvol%bshare      share.smb.rw                                    default
```

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 `cshare`. The second command adds the `csc` property.

```
$ zfs share -o share.smb=on tank/zvol3%cshare
$ zfs set -o share.smb.csc=auto tank/zvol3%cshare
```

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/engadmin`:

```
*       /export/home/&    share.smb.csc=disabled,description=&
userone    /export/home/&    share.smb.csc=auto,dn=oracle,dn=com,ou=users
```

Example 3-7 Using `ls` and `chmod` to Manage SMB Share-Level ACLs

Although you can manage share ACLs on an Oracle Solaris system, a better practice is 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. For more information about using the `chmod` command to modify ACLs, see the `chmod`(1) man page.

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 about 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, `userone`, 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 `userone`. Note that the ACL entry for `everyone` is unchanged.

```
$ chmod A+user:userone: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:userone: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 `userone` to permit all access privileges. Now, the `ls -lv` command shows that the ACL entry for user `userone` has been updated to have all access privileges.

```
$ chmod A0=user:userone: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:userone: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
```

Enabling Guest Access

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. By default, guest access is disabled. To enable guest access set the `guestok` property to `on`.

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 although 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 Guest Access to an SMB Share

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`(8) man page.

1. **Become an administrator.**

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

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

```
$ zfs create -o mountpoint=/eng pool/eng
$ zfs share -o share.smb=on -o share.smb.guestok=on pool/eng%eshare
```

Example 3-8 Enabling Guest Access to an SMB Share

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

```
$ zfs share -o share.smb=on -o share.smb.guestok=on tank/home%myshare
```

Enabling 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 the following benefits:

- Finding files in directories that contain many files is easier because the number of files shown in the listing is reduced.
- 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_share`(8) man page.

ABE filtering is also supported on autohome shares. See the `smbautohome`(5) man page.

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

> **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.

How to Enable Access-Based Enumeration for a Share

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`(8) man page.

1. **Become an administrator.**

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

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

   ```
   $ zfs share -o share.smb=on -o share.smb.abe=on pool/dataset%share-name
   ```

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

   ```
   $ zfs create tank/home
   $ zfs share -o share.smb=on -o share.smb.abe=on tank/home%myshare
   ```

How to Modify SMB Share Properties (`zfs`)

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`(8) man page.

1. **Become an administrator.**

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

2. **View the existing share.**

   ```
   $ zfs get share.all,share.smb.all tank/home%home
   NAME            PROPERTY                VALUE       SOURCE
   tank/home%home  share.desc                          default
   tank/home%home  share.name              home        -
   tank/home%home  share.nfs               off         default
   tank/home%home  share.nfs.*             ...         default
   tank/home%home  share.path                          default
   tank/home%home  share.point             /tank/home  -
   tank/home%home  share.protocols         smb         local
   tank/home%home  share.smb               on          local
   tank/home%home  share.smb.*             ...         default
   tank/home%home  share.state             shared      -
   tank/home%home  share.smb.abe           off         default
   tank/home%home  share.smb.ad-container              default
   tank/home%home  share.smb.catia         off         default
   tank/home%home  share.smb.cont_avail    off         default
   tank/home%home  share.smb.csc                       default
   tank/home%home  share.smb.dfsroot       off         default
   tank/home%home  share.smb.guestok       off         default
   tank/home%home  share.smb.none                      default
   tank/home%home  share.smb.ro                        default
   tank/home%home  share.smb.rw                        default
   ```

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

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

   ```
   $ zfs set share.smb.guestok=off tank/home%home
   ```

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

```
$ zfs set share.smb.csc=disabled tank/home%home
```

For information about available SMB share properties, see the `share_smb`(8) man page.

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`(8) man page.

1. **Become an administrator.**

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

2. **Remove an SMB share.**

   ```
   $ zfs destroy pool/dataset%share-name
   ```

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

   ```
   $ zfs destroy tank/sales%share_sales1
   ```

Creating an 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.

How to Create a Specific Autohome Share Rule

This procedure describes how to configure autohome shares by adding rules to a configuration file.

For information about the `smbautohome` format, see SMB Autohome Entries and the `smbautohome`(5) man page.

1. **Become an administrator.**

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

2. **Add an autohome entry to the `/etc/smbautohome` file.**

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

   ```
   key
   location [container]
   ```

key

Usually 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.

location

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/auser`:

```
auser             /home/?/&
```

For more information about the path, see SMB Autohome Shares.

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*.

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

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

For more information about the access control mechanisms that are used for shares, see Host-Based Access Control to SMB Shares.

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`(8) man page.

For information about access lists, see the `share_smb`(8) man page.

1. **Become an administrator.**

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

2. **Determine the type of access you want to grant for each client host.**
3. **Restrict access by particular hosts to a share.**

```
$ zfs share -o share.smb=on -o share.smb.ro=hostname[:hostname] pool/dataset%share-name
$ zfs share -o share.smb=on -o share.smb.rw=hostname[:hostname] pool/dataset%share-name
$ zfs share -o share.smb=on -o share.smb.none="" pool/dataset%share-name
```

hostname

A host name, a netgroup, or an IP address

pool/dataset%share-name
Name of the dataset and share being shared

You can specify the host access policy by combining the access settings in a single command.

Example 3-9 Setting Host Access Policy by Using a Single Command

The following command specifies how particular hosts can access the `acme.sales.logs` share. The `mercury` and `venus` hosts have read-write access, `mars` has read-only access, and `neptune` has no access.

```
$ zfs share -o share.smb=on -o share.smb.rw=mercury:venus,ro=mars,none="*" \
tank/sales/logs%acme.sales.logs
```

Managing SMB Groups

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

For information about SMB groups and local users, see Local SMB Groups.

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

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, you must create an Oracle Solaris group before you can create an SMB group. You create the Oracle Solaris group by using the `groupadd` command. See the `groupadd`(8) man page.

Task	Description	For Instructions
Create an SMB group.	Create an SMB group to manage users.	How to Create an SMB Group
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
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

Task	Description	For Instructions
Modify SMB group properties.	An SMB group can grant the following privileges: • `backup` – Permits group members to back up file system objects. • `restore` – Permits group members to restore file system objects. • `take-ownership` – Permits group members to take ownership of file system objects. You can specify a description of the SMB group by modifying the value of the `description` property.	How to Modify SMB Group Properties

How to Create an SMB Group

Before You Begin

You must create an Oracle Solaris group before you can create an SMB group. For more information, see How to Add a Group in *Managing User Accounts and User Environments in Oracle Solaris 11.4*.

1. **Become an administrator.**

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

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.**

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

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 `userone` of the `sales` domain to the `wsales` group, type:

```
$ smbadm add-member -m sales\\userone wsales
```

You add a local user to an SMB group in workgroup mode by specifying the Oracle Solaris user name. For example, to add local user `userone` of the `solarsystem` host to the `wsales` group, type:

```
$ smbadm add-member -m userone wsales
```

How to Remove a Member From an SMB Group

1. **Become an administrator.**

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

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

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

   member-name can be specified in the following ways:

   ```
   [domain-name\]username [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 `userone` of the `sales` domain from the `wsales` group, type:

   ```
   $ smbadm remove-member -m sales\\userone wsales
   ```

   You can remove a local user from an SMB group in workgroup mode by specifying the Oracle Solaris user name rather than the domain name. For example, to remove local user `userone` of the `solarsystem` host from the `wsales` group, type:

   ```
   $ smbadm remove-member -m solarsystem\\userone wsales
   ```

How to Modify SMB Group Properties

1. **Become an administrator.**

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

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
   ```

   For more information about available group properties, see the `smbadm`(8) man page.

Configuring the WINS Service

This section provides information about configuring the SMB server as a client to the WINS service. Configuring the WINS service enables the system to use the NetBIOS name service.

If you are integrating an SMB server in an environment that has a WINS server, you can use WINS for name resolution.

For information about excluding IP addresses from WINS resolution, see Excluding IP Addresses From WINS Name Resolution.

For information about configuring other applicable services, see Configuring the SMB Server – Process Overview.

How to Configure WINS

Before You Begin

> **Note:**
>
> The `wins_server_1` and `wins_server_2` properties are not needed if NetBIOS is disabled. See Disabling and Re-enabling NetBIOS.

1. **Become an administrator.**

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

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
   ```

Disabling and Re-enabling NetBIOS

NetBIOS provides the following three services:

Name Service
The name service is for name registration and resolution and utilizes UDP port 137.

Datagram Service
The datagram service is for connectionless registration communication over a network. It allows for message broadcasts to all computers on a network and receipt of mailslot messages in order to locate domain controllers via NetBIOS-based discovery. The datagram service utilizes UDP port 138.

Session Service
The session service enables two computers to establish a connection for SMB traffic. The session service utilizes UDP port 139.

NetBIOS services are enabled by default. If you want to disable NetBIOS services, set the value of the `netbios_enable` property to `false`.

You should disable NetBIOS services if all of the following conditions are met:

- No SMB clients use NetBIOS over TCP/IP (NBT) as the transport layer.
- NetBIOS is not used for name resolution. NetBIOS can be used for name resolution if DNS is not supported or is not working on the local network.
- NetBIOS is not used for domain controller discovery.

By default, the first label of the fully qualified AD domain name is the NetBIOS domain name. If a non-default NetBIOS domain name is specified for AD domain join, then Oracle Solaris will perform NetBIOS-based domain controller discovery as part of the domain join process. To eliminate the dependency on NetBIOS-based domain controller discovery, use fully qualified AD domain names.

The `netbios_enable` property enables or disables all NetBIOS services. A value of `true` (the default) enables NetBIOS name (UDP port 137), datagram (UDP port 138), and session (TCP port 139) services and enables the ability to locate domain controllers via NetBIOS-based discovery. A value of `false` disables all of these services and capabilities.

Use the `sharectl` command to change the value of the `netbios_enable` property. The default value is `true`:

```
$ sharectl get -p netbios_enable smb
netbios_enable=true
```

To disable NetBIOS services, set the value of the `netbios_enable` property to `false`:

```
$ sharectl set -p netbios_enable=false smb
$ sharectl get -p netbios_enable smb
netbios_enable=false
```

To re-enable NetBIOS services, set the value of the `netbios_enable` property to `true`:

```
$ sharectl set -p netbios_enable=true smb
$ sharectl get -p netbios_enable smb
netbios_enable=true
```

Enabling CATIA V4/V5 Character Translations

The CATIA V4 product is available only 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 required only 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

You use the `zfs` command as an administrator 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.

To enable CATIA translation for a share, type the following command:

```
$ zfs set share.smb.catia=true pool/dataset
```

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

```
$ zfs set share.smb.catia=true files/acme.sales.logs
```

Troubleshooting the SMB Service

This section describes some troubleshooting issues for the Oracle Solaris SMB service. For related troubleshooting information, see the following:

- Troubleshooting the Identity Mapping Service
- Troubleshooting the SMB Client

Cannot Join a Windows Domain

To authenticate users from a Windows domain, the Oracle Solaris SMB service must locate a domain controller, authenticate, and then add a computer account to the domain.

Users from the domain are not able to establish a connection to the Oracle Solaris SMB service unless this process succeeds.

Check the DNS Configuration

The Oracle Solaris SMB service must be running for the `smbadm join` command to succeed.

If Active Directory (AD) is configured, the Oracle Solaris SMB service attempts to locate the domain controller by means of DNS. If the service cannot locate the domain controller, you must use SMF to configure DNS properly.

The following configuration issues might prevent you from configuring the Oracle Solaris SMB service in domain mode:

- **Missing DNS domain.** Ensure that the fully qualified AD domain name has been added to the search list or as the local domain.

  If your configuration is incorrect, you might see the `Failed to join domain` *`domain-name`* `(INVALID_PARAMETER)` error when attempting to join the domain.

- **Missing DNS server.** Ensure that the IP address of the AD DNS server is added as the name server.

  If your configuration is incorrect, you might see the `Failed to find any domain controllers` error when attempting to join the domain.

- **DNS host lookup not used.** Ensure that DNS is used for host lookup.

Use the `svccfg` command to update properties for `system/name-service/switch` and `network/dns/client`. See the `svccfg`(8) man page.

Unable to Access SMB Shares from Windows 10 Client When TryIPSPN Registry Is Enabled

Starting with Windows 10, clients can enable Kerberos authentication with IP address-based Service Principal Names (SPNs) by creating a TryIPSPN registry entry. However, Oracle Solaris does not support IP address-based SPNs.

Like Windows, Oracle Solaris does not register IP address-based SPNs as part of the AD domain join. Microsoft notes the following potential issues with IP address-based SPNs: IP addresses are not normally used in place of hostnames because IP addresses are often temporary. Using IP addresses can lead to conflicts and authentication failures as address leases expire and renew. Therefore, registering an IP address-based SPN is a manual process and should only be used when it is not possible to switch to a DNS-based hostname.

When the TryIPSPN registry setting is enabled, Windows 10 clients can continue to access SMB shares exported by the Oracle Solaris system via NTLMSSP. If the domain administrator has manually added `cifs\`*`IP-address`* SPNs to the

`ServicePrincipalName` attribute of the AD computer object, Windows KDC will be able to issue CIFS service tickets for the Oracle Solaris system using IP address-based SPNs. However, if the client subsequently presents a CIFS service ticket that uses IP address-based SPNs to access SMB resources on the system, the Oracle Solaris Kerberos subsystem will reject the security context due to the lack of support for IP address-based SPNs. As a result, `syslog` will contain the following notice-level messages:

```
krbssp: user authentication failed (GSS major error): No credentials were
supplied, or the credentials were unavailable or inaccessible
krbssp: user authentication failed (GSS minor error): No principal in keytab
('FILE:/var/krb5/krb5.keytab') matches desired name cifs/IP-address@AD-realm
```

To avoid this issue, do not publish IP address-based SPNs to an AD server.

Ensure That You Specify the Correct Password for Your Domain User

The user that you specify on the `smbadm join` command line must have the correct password and the authority to create computer accounts.

The following error message appears if you supply the wrong password for the administrative user:

```
Failed to find any domain controllers for domain-name
```

Ensure the Firewall Software Does Not Filter Out Required Ports

Some firewall software might filter out certain ports, which will prevent an SMB server from successfully joining a domain.

The following network protocols are used by the `smbd` service during a domain join operation, and must be available for the Oracle Solaris SMB service:

Domain Name Service (DNS)
53

Kerberos V Authentication
88

Kerberos V Change & Set Password (`RPCSEC_GSS`)
749

LDAP
389

NetBIOS Name Service
137

NetBIOS Datagram
138

SMB-over-NetBIOS
139

SMB-over-TCP
445

> **Note:**
>
> Ports 137, 138, and 139 are not required if NetBIOS services are disabled. See Disabling and Re-enabling NetBIOS.

Port assignment settings appear in the `/etc/services` file. For more information, see the `services`(5) man page.

Verifying Oracle Solaris SMB Service Property Settings

Much of the Oracle Solaris SMB service configuration uses the `sharectl`(8) command to set properties. Before you change property values, you should view the current property settings by running the `sharectl get smb` command.

Troubleshooting SMB Server Issues

For troubleshooting SMB server issues, it is helpful to enable debug level messages. This action however generates a large number of messages. So, it is advised to disable debug messages when done with troubleshooting.

The SMF log for the SMB Service is located in `/var/svc/log/network-smb-server:default.log`.

Excluding IP Addresses From WINS Name Resolution

When using WINS/NetBIOS, Windows domain controllers (DC) do not automatically respond to the host from which they received a request. They perform a WINS or NetBIOS cache lookup and for multihomed servers, the DC can respond to different network interfaces belonging to the server. If the IP address is not accessible to the DC, it will appear as if the DC has not responded to the server. Therefore, you might have to exclude specific network interfaces from WINS registration.

The following example shows how to configure the Oracle Solaris SMB service as a WINS client. The primary WINS server is set to IP address `192.0.2.20`, the secondary WINS server is set to IP address `192.0.2.21`, and network interfaces `bge0` and `bge1` are excluded from WINS resolution.

```
$ sharectl set -p wins_server_1=192.0.2.20 smb
$ sharectl set -p wins_server_2=192.0.2.21 smb
$ sharectl set -p wins_exclude=bge0,bge1 smb
```

Changes to Windows Group Membership and to User Mapping Do Not Take Effect

Windows clients use an access token to assign user data and group membership. This token is assigned when the client connects to the SMB service. Any changes made to this token are not reflected until the next time the user connects.

To force changes to take effect immediately, the user must disconnect from the SMB service by logging out of all connected workstations.

Windows Clients Cannot Connect by NetBIOS Name or Are Missing From the Browse List or Network Neighborhood

A master browser is a server that is configured to manage SMB browse lists and to respond to client requests for them. A Windows server is configured as a master browser by default.

The Oracle Solaris SMB service is not configured as a master browser. The Oracle Solaris SMB service dedicates all of its resources to file sharing.

For browsing to function correctly, each subnet or physical network segment must have a master browser. To make the Oracle Solaris SMB service available through browse lists, the system on which it runs should be located on the same segment and subnet as a Windows server.

Configuring a Windows server improves the performance of browsing and might compensate for the lack of a master browser on some segments.

Cannot Set Share Security; All Shares Inherit the Security of the Directory Object

The security implementation of the Oracle Solaris SMB service secures only files and directories. The effective security of a SMB share is *always* the security of the directory to which it points.

Older Versions of Windows Cannot Copy Files Larger Than Four Gbytes

You might see this problem if your client is running Windows 2000 or an older version of Windows.

- If you are running a Windows 2000 client, apply the latest service pack.
- If you are running a version older than Windows 2000, you might be able to work around the problem by using the Windows backup utility or by using a similar third-party product.

Cannot Use SMB to Map Drives

To map a drive or to connect to a share, you must have read access to the directory to which the share points.

If the Oracle Solaris SMB service is in domain mode, you must also be logged in to the domain.

To ensure that a user can connect to a share, do the following steps to check and modify permissions:

1. Log in to the system that is running the Oracle Solaris SMB service.
2. Become superuser.
3. Obtain the user name and group name of the owner.

   ```
   $ ls -l pathname
   ```

   For example, the following output indicates that the share is a directory with 750 permissions. The owner is `root` and the group is `sys`.

```
$ ls -ld /vol1/data
drwxr-x---  41 root     sys         1024 Jan  2 23:19 /vol1/data
```

4. Determine the permissions necessary for the user to access the directory.
5. Use the `chmod` command to change the permissions of the directory.

Microsoft Access or SQL Server Sessions Time Out After a Period of Inactivity

Applications can send SMB echo requests periodically to keep idle sessions open or to reconnect, as required, if a session times out due to inactivity. If an application appears unable to deal with an idle session timeout, the SMB service `keep_alive` property can be set to 0 to disable the session inactivity timer.

```
$ sharectl set -p keep_alive=0 smb
```

Cannot Add Windows Local Groups to Access Control List

You cannot use Windows local groups to assign security on remote systems. You can use local group only on the individual computer on which it is created. A local group is not stored in the domain SAM database.

Windows domain controllers are an exception to this behavior. Domain controllers share a set of local groups that can be shared only with other domain controllers. To make security assignments to the Oracle Solaris SMB service, use global groups.

The Oracle Solaris SMB service has its own set of local groups that are provided for Windows compatibility purposes. These local groups permit a limited set of privileges, and they can also be used for security assignments to individual files and folders.

> **Note:**
>
> Windows domain local groups are not supported.

SMB Browsing Fails When `share.smb=on` Is Set on a ZFS Pool

If you have a ZFS pool with datasets and you run the `zfs set share.smb=on` command on the pool, the pool and all its datasets are shared but unavailable for browsing by Windows systems.

To work around this problem:

1. Determine whether your ZFS pool and dataset versions support SMB shares.

   ```
   $ zpool get version pool
   $ zfs get version dataset
   ```

   Support for SMB shares requires that ZFS pools be at least Version 9 and that ZFS datasets be at least Version 3.
2. If your ZFS pool or dataset version are out of date, upgrade your ZFS pools and datasets.

```
$ zpool upgrade pool
$ zfs upgrade dataset
```

For more information, see the zpool(8) and zfs(8) man pages.

3. Either map the shares directly or run the `zfs set share.smb=on` command on any of the lower-level datasets instead of the pool.

Samba or SMB Service Cannot Bind Various Ports

Errors will occur if you attempt to run both the Samba service `svc:/network/samba:default` and the Oracle Solaris SMB service `svc:/network/smb/server:default` simultaneously.

The Samba and Oracle Solaris SMB services are mutually exclusive because they both attempt to listen on the same ports. Only one service should be enabled at any time.

To disable the Samba service, use the `svcadm disable svc:/network/samba` command.

To disable the Oracle Solaris SMB service, use the `svcadm disable smb/server` command.

Invalid Password Errors Appear When Mapping a Drive or Browsing Computers in the Workgroup

When you map a drive or browse computers in your workgroup, you might see `invalid password` errors. If you see these errors, check whether the `/var/smb/smbpasswd` file includes information for the appropriate users.

Also, ensure that the `pam_smb_passwd.so.1` entry is in the `/etc/pam.d/other` file and that you use the `passwd` command to set your password.

For more information, see How to Configure the SMB Server in Workgroup Mode.

Access Control List Inheritance Issues

Access control list (ACL) behavior differs between Windows systems and ZFS file systems on Oracle Solaris systems. You might experience Windows ACL inheritance problems because of the access control entry (ACE) ordering used by the default ZFS ACL.

The default ZFS ACL is designed to comply with POSIX, which results in the interleaving of `allow` and `deny` ACEs. Windows expects all `deny` ACEs to precede all `allow` ACEs.

You can override the default ZFS behavior by changing the ACL on the root directory to provide the equivalent of `Everyone:FullControl` as follows:

```
$ chmod 777 /pool-name
$ chmod A=everyone@:rwxpdDaARWcCos:fd:allow /pool/dataset
```

For information about the `chmod` options, see the chmod(1) man page.

You can verify the ACL by viewing it on Windows or by running the following command on an Oracle Solaris system:

```
$ ls -V -d /pool/dataset
```

You can apply this ACL recursively to all subdirectories and files for existing file systems from Windows or from the Oracle Solaris OS.

If you apply the ACL when the file system is first created, the ACL will be propagated according to the normal inheritance rules.

If a directory has a default ZFS ACL, when a file or folder is created in this directory from Windows, it has two ACEs: one for the owner and one for SYSTEM. To change this behavior, update the root directory's ACL by running the `chmod` commands.

Cannot See the Security Tab From Windows Clients

Some Windows clients do not show the security tab unless you have permission to view or change security.

For information about how to view and modify share permissions, see Cannot Use SMB to Map Drives.

Missing Security Tab on Windows XP Clients

You might not see the security tab for a file or folder when using an XP client for the following reasons:

- You do not have enough permissions to see the security settings of the file or folder
- Simplified file sharing is enabled on your client

To disable simplified file sharing, choose Control Panel → Folder Options → View, deselect Use Simple File Sharing (Recommended), and click Apply.

For more information about disabling simplified file sharing and setting permissions on a shared folder, see Microsoft knowledge base article 307874.

4

Using SMB File Sharing on Client Systems

You can use an SMB client on an Oracle Solaris system to access SMB shares from both Oracle Solaris and Windows systems. This chapter describes how to use the SMB client as an unprivileged and as a privileged user to access SMB shares.

This chapter covers the following topics:

- Managing SMB Mounts in Your Local Environment
- Managing SMB Mounts in the Global Environment
- Troubleshooting the SMB Client

Managing SMB Mounts in Your Local Environment

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 Join a Kerberos Client to an Active Directory Server in *Managing Kerberos in Oracle Solaris 11.4*
Find the shares that are available on an SMB server in your domain.	View the shares from a particular SMB server, which you can mount on a directory that you own.	How to Find Available SMB Shares on a Known File Server
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
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
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
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.	Storing SMB Persistent Passwords
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.	Configuring the PAM Module to Store an SMB Persistent Password

Task	Description	For Instructions
Delete a persistent password.	If you no longer want to store a persistent password, delete it.	Deleting an SMB Persistent Password

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

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

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

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

   `-A`

   Enables you to view shares anonymously

   `-u`

   Indicates the user to authenticate on the specified SMB server

   `-t`

   Displays 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.

4. **If prompted, provide the password for the user that you specified on the SMB server.**

5. **View the list of available SMB shares.**

   Use the `smbadm show-shares -t` command to display the names and text descriptions of the shares with output headers.

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

```
$ smbadm show-shares -t -A 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 view the shares on the `solarsystem` server without the output headers:

```
$ 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
   ```

   `-A`
   Enables you to view shares anonymously

   `-u`
   Indicates the user to authenticate on the specified SMB server

   `-t`
   Displays 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. **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
   ```

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

   ```
   $ mount -F smbfs [-o dirperms=octal-triplet,fileperms=octal-triplet,gid=group-ID,\
   uid=user-ID,user=username,...] //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
   ```

   You can use the following options to set the directory access permissions in the client:

- `dirperms=`*octal-triplet* – Specifies the directory permissions that you can set to the directories. The `dirperms` permission does not affect the access policies that the SMB server maintains. By default, the system uses the value that you set for the `fileperms`. The system then adds `execute` permissions to the `fileperms` settings.

  For example, you can set `dirperms` to `700` to prevent group members from accessing the directory.

  ```
  $ ls -ld /export/home/user1/mnt
  drwxr-xr-x 2 user1 staff 2 Dec 16 13:57 /export/home/user1/mnt
  $ mount - F smbfs -o dirperms=770 //server/share /export/home/user1/mnt

  $ su - user2
  user2% gid -gn
  staff
  user2% ls -ld /export/home/user1/mnt/test
  drwxrwx---+ 1 user1 staff 512 Aug 27 14:58 /export/home/user1/mnt/test
  user2% exit

  $ umount /export/home/user1/mnt
  $ mount - F smbfs -o dirperms=700 //server/share /export/home/user1/mnt

  $ su -user2
  user2% ls -ld /export/home/user1/mnt/test
  /export/home/user1/mnt/test:Permission denied
  ```

- `fileperms=`*octal-triplet* – Specifies the file permissions that you can set to the files on a mount point. The `fileperms` permission does not affect the access policies that the SMB server maintains. By default, the file permission is `700`.

  For example, you can set `fileperms` to `770` to enable the group associated with the mount point to access files on the mount point.

  ```
  $ ls -ld /export/home/user1/mnt
  drwxr-xr-x 2 user1 staff 2 Dec 16 13:57 /export/home/user1/mnt
  $ mount - F smbfs //server/share /export/home/user1/mnt

  $ su - user2
  user2% gid -gn
  staff
  user2% ls -l /export/home/user1/mnt/test.file
  /export/home/user1/mnt/test.file: Permission denied
  user2% exit

  $ umount /export/home/user1/mnt
  $ mount - F smbfs -o fileperms=770 //server/share /export/home/
  user1/mnt

  $ su - user2
  user2% ls -l /export/home/user1/mnt/test.file
  -rwx------+ 1 user1 staff 0 Dec 16 15:56 /tmp/mnt/test.file
  ```

- `gid=`*group-ID* – Specifies the group ID that you can set as the effective group. The effective group uses the group permissions that is set for the mount point. By default, the value of `gid` is the group ID of the mount point.

  The following example shows that the group ownership of the mount point `/export/home/user1/mnt` is set to `staff`. You can set the `gid` to `not_staff` to enable the users in the `not_staff` group to access the files on the mount point.

```
$ ls -ld /export/home/user1/mnt
drwxr-xr-x 2 user1 staff 2 Dec 16 13:57 /export/home/user1/mnt
$ mount -F smbfs -o dirperms=770 //server/share /export/home/user1/mnt

$ su -user2
user2% gid -gn
not_staff
user2% ls -ld /export/home/user1/mnt/test
/export/home/user1/mnt/test: Permission denied
user2% exit

$ umount /export/home/user1/mnt
$ mount -F smbfs -o gid=not_staff,dirperms=770 //server/share /export/home/
user1/mnt

$ su - user2
user2% ls -ld /export/home/user1/mnt/test
drwxr-x---+ 1 user1 staff 512 Dec 16 15:56 /tmp/mnt/test
```

- `uid`=*user-ID* – Specifies the local user ID that you can set as the effective owner. The effective owner uses the owner permissions that is set for the mount point. By default, the value of `uid` is the user ID of the mount point.

  For example, to enable `user2` to access the mounted files, you can set `uid` to `user2`.

```
$ ls -ld /export/home/user1/mnt
drwx------ 2 user1 staff 2 Dec 16 13:57 /export/home/user1/mnt
$ mount - F smbfs //server/share /export/home/user1/mnt

$ su - user2
user2% ls -ld /export/home/user1/mnt/test
/export/home/user1/mnt/test: Permission denied
user2% exit

$ umount /export/home/user1/mnt
$ mount - F smbfs -o uid=user2 //server/share /export/home/user1/mnt

$ su - user2
user2% ls -ld /export/home/user1/mnt/test
drwxr-x---+ 1 user1 staff 512 Dec 16 15:56 /tmp/mnt/test
```

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:

```
$ 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. You can also use the `df` command with the `-F smbfs` option to list the SMB mounts.

```
$ 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:

   - ```
     $ 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
     ```

   - ```
     $ 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
   ```

About Persistent Passwords

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.

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:**
>
> 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.

Storing SMB Persistent Passwords

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.

To store the persistent password for the SMB server, type the following command:

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

You can specify the user name as a single name or use a format such as *domain\username* or *username@domain*.

The following command stores the persistent password for `user1@solarsystem`. Each time `user1` performs a transaction with `solarsystem`, the persistent password is used to perform the authentication.

```
$ smbadm add-key -u user1@solarsystem
Password for SOLARSYSTEM/user1:
```

Configuring 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.

Use your login name and password to store a persistent password.

Add the following line to the end of the `/etc/pam.d/login` file:

```
auth optional           pam_smbfs_login.so.1
```

This action adds a persistent password entry whenever a user logs into the system, as if they 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.

For more information, see the `pam_smbfs_login`(7) man page.

Deleting an SMB Persistent Password

You can delete persistent passwords that are stored by the `smbadm add-key` command.

To delete a single persistent password that was created by the user running the `smbadm remove-key` command, type the following command:

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

For example, the following command removes the persistent password for `user1@solarsystem`:

```
$ smbadm remove-key -u user1@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 `user1` 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

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 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.

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 users mounting the share in their own directories, 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
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
View the SMB property values.	Use the `sharectl` command to view SMB property values.	How to View the SMB Environment Property Values
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

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 a superuser, you do not need to own the mount point. Mount options control the access to the mount point. You access the server as the user who mounted the share.

1. **Become an administrator.**

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

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. **Perform the mount.**

```
$ mount -F smbfs [-o user=user-name,dirperms=octal-triplet,fileperms=octal-
triplet,\
gid=group-ID...] //server/share mount-point
```

   user-name

   Specifies the account used to authenticate the user when accessing a remote system.

   octal-triplet

   Specifies the directory permissions that you can set to the directories. The `dirperms` permission does not affect the access policies that the SMB server maintains.

   octal-triplet

   Specifies the file permissions that you can set to the files on a mount point. The `fileperms` permission does not affect the access policies that the SMB server maintains.

   group-ID

   Specifies the group ID that you can set as the effective group. The effective group uses the group permissions that is set for the mount point.

Example 4-1 Mounting a Multiuser SMB Share

In this example, `sales-tool` share is mounted at the `/sales` mount point. This mount point is owned by the user `user1` and group `salesgrp`. Mount options enable read and write access to the users belonging to the `salesgrp` group.

Use the `smbadm show-shares` command to list the shares.

```
$ smbadm show-shares -A solarsystem
c$      Default Share
IPC$    Remote IPC
sales-tools
```

Mount the `sales-tools` share to `/sales` mount point.

```
$ mkdir -m 770 /sales
$ chown user1:salesgrp /sales
$ ls -ld /sales
drwxrwx---   2 user1    salesgrp    117   Feb 17 13:24 /sales
$ mount -F smbfs -o user=user1,fileperms=770,dirperms=770,gid=salesgrp \
//solarsystem/sales-tools /sales
```

Mount options enable the user `user1` to access the `sales-tools` share on the system `solarsystem`. These options also enable users in the `salesgrp` group to access the files and directories. User `auser` can access the share as the user belongs to the `salesgrp` group. However, any access to the mount point such as creating a file in the mount point can be done only by the user `user1`.

```
$ su -auser
% id -gn salesgrp
% cd /sales
% ls -l
total 0
drwxr-x---+  1 auser    salesgrp     512 Feb 17 14:22 central
-rwxr-----+  1 user1    salesgrp       0 Feb 17 14:22 contacts
drwxr-x---+  1 user1    salesgrp     512 Feb 17 14:22 east
-rwxr-----+  1 buser    salesgrp       0 Feb 17 14:22 numbers
drwx------+  1 cuser    fingrp       512 Feb 17 14:22 west
% touch my-file
% ls -l my-file
-rwxrwx---+  1 user1    salesgrp       0 Feb 17 14:34 my-file
```

A user who does not belong to the `salesgrp` group cannot access the mount point.

```
$ su - cuser
% id -gn
fingrp
% cd /sales
cd: /sales: [Permission denied]
```

You can remount the share using the `uid` mount option to enable the user `cuser` to access the share.

```
$ umount /sales
$ mount -F smbfs -o user=user1,fileperms=770,dirperms=770,gid=salesgrp,uid=cuser\
   //solarsystem/sales-tools /sales
$ su - cuser
% id -un
buser
% cd /sales
% ls -l
-rwxrwx---+  1 user1    salesgrp       0 Feb 17 14:34 my-file
drwxr-x---+  1 auser    salesgrp     512 Feb 17 14:22 central
-rwxr-----+  1 user1    salesgrp       0 Feb 17 14:22 contacts
drwxr-x---+  1 user1    salesgrp     512 Feb 17 14:22 east
-rwxr-----+  1 buser    salesgrp       0 Feb 17 14:22 numbers
drwx------+  1 cuser    fingrp       512 Feb 17 14:46 west
```

The user `cuser` who belongs to the `fingrp` group owns the `west` directory in the `/sales` mount point. However, user `cuser` cannot access the `west` directory, as the `/sales` mount point is mounted by user `user1` who does not belong to the `fingrp` group.

```
% ls -l west
ls: error reading directory west: Permission denied
```

How to Customize the SMB Environment in Oracle Solaris

1. **Become an administrator.**

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

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

   For a description of the properties, see the `smb`(5) 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`(8) command.

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

  For a description of the properties, see the `smb`(5) 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 Storing SMB Persistent Passwords.

> **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 be used only 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 Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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. **Add the mapping to the automount map.**
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.

Example 4-2 Editing the automounter map

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.

Troubleshooting the SMB Client

The following are troubleshooting issues for the Oracle Solaris SMB client. For related troubleshooting information, see the following:

- Troubleshooting the Identity Mapping Service

- Troubleshooting the SMB Service

Viewing SMB Client Property Settings

The Oracle Solaris SMB client configuration uses the `sharectl` command to set properties. Before you change property values, use the `sharectl get smb` command to view the current property settings.

Access Denied Message When Accessing a Server

If you get an `Access Denied` error when attempting to access or view SMB shares from a server, the password you supplied might be wrong or the SMB server might be part of a different domain.

If the SMB server and SMB client are in different domains, you must provide the domain name for the `smbadm show-shares` or `mount` command. Otherwise, the server assumes that you are attempting to authenticate a local user, and the authentication process fails.

For example, if the server `solarsystem` is in the `mydomain` domain, the following commands would be appropriate to view and access SMB shares as user `auser`:

```
$ smbadm show-shares -u "auser@mydomain" solarsystem
$ mount -F smbfs -o user=auser,domain=mydomain //solarsystem/tmp /mnt
```

Cannot View or Mount SMB Shares

If you are unable to view or mount SMB shares, use the following command:

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

The `-A` option gives anonymous access to the server if the server permits such access.

Files and Directories in Multilevel Shares are Inaccessible

If the `zfs multilevel` property is set to `on`, then, only files and directories with the default label, `ADMIN_LOW`, are accessible to SMB clients.

Cannot Mount SMB Shares as a Regular User

You might see the following error message when you attempt to mount an SMB share as a regular user on a mount point that you own:

```
$ mount -F smbfs //server-name/share-name mount-point
Insufficient privileges, mount must be set-uid root
```

Verify that you have the following entries in your `/etc/security/exec_attr.d/core-os` file:

```
Forced Privilege:solaris:cmd:RO::/usr/lib/fs/smbfs/mount:privs=sys_mount
Forced Privilege:solaris:cmd:RO::/usr/lib/fs/smbfs/umount:privs=sys_mount
```

These entries in the `/etc/security/exec_attr.d/core-os` file enable you to mount and unmount SMB shares on mount points that you own as a regular user.

If you need to specify a user name, use the `-o` option:

```
$ mount -F smbfs -o user=username //server-name/share-name mount-point
```

tar and gtar Warnings

You might see the `File changed as we read it` warning in the following situations:

- When you use the Oracle Solaris SMB client to mount an SMB share and use the `gtar` utility to write the share to a tape
- When you use the Oracle Solaris SMB client to mount an SMB share and use the `tar` utility to check file attributes after setting them

Other than these warnings, the `tar` and `gtar` operations succeed as expected.

You can ignore these warnings.

> **Note:**
>
> `smbfs` ignores calls to set any file or directory attributes because they have no direct representation in SMB. Also, `smbfs` does not support the UNIX extensions that would permit the storing of attributes with some servers.

Viewing XATTR Status for Mounted Shares

By default, shares that are mounted by the `mount_smbfs` command enable Oracle Solaris extended attributes by setting the `xattr` mount option. However, if the SMB server does not support Windows named streams, shares mounted by `mount_smbfs` set the `noxattr` mount option.

To verify whether the `xattr` or `noxattr` mount option is used, type the following command:

```
$ mount -v | grep 'type smbfs'
```

The following example shows that the share mounted on `/mnt` has `xattr` set, while the share mounted on `/tmp` has `noxattr` set:

```
$ mount -v | grep 'type smbfs'
//root@solarsystem/tmp on /mnt type smbfs
  remote/read/write/setuid/devices/intr/xattr/dev=5080000 on Tue Jun  5 18:20:48
2012
//root@pluto/files on /files type smbfs
 remote/read/write/setuid/devices/intr/noxattr/dev=4800000 on Mon Jun  4
11:37:26 2012
```

A

SMB DTrace Provider

This appendix provides information about the SMB and SMB2 DTrace provider, which enables you to use stable probe names to write DTrace scripts for the SMB and SMB2 server. DTrace is a feature of the Oracle Solaris 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 *Oracle Solaris 11.4 DTrace (Dynamic Tracing) Guide* and the `dtrace`(8) 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

DTrace provides the SMB DTrace probes and SMB2 DTrace probes for debugging SMB calls. Most probes do not take an argument.

To list the available DTrace probes, type the following commands:

- For SMB DTrace probes, run the `dtrace -P smb -l` command.
- For SMB2 DTrace probes, run the `dtrace -P smb2 -l` command.

The DTrace `Read` and `Write` probes take an `arg[2]` argument.

`smbReadArgs_t *` is the `arg[2]` argument for the following SMB and SMB2 DTrace `Read` probes:

```
smb:::op-Read-start
smb:::op-Read-done
smb:::op-ReadRaw-start
smb:::op-ReadX-start
smb:::op-ReadX-done
smb:::op-LockAndRead-start
smb:::op-LockAndRead-done
```

```
smb2:::op-Read-start
smb2:::op-Read-done
```

`smbWriteArgs_t *` is the `arg[2]` argument for the following SMB and SMB2 DTrace `Write` probes:

```
smb:::op-Write-start
smb:::op-Write-done
smb:::op-WriteAndClose-start
smb:::op-WriteAndClose-done
smb:::op-WriteAndUnlock-start
smb:::op-WriteAndUnlock-done
smb:::op-WriteRaw-start
smb:::op-WriteRaw-done
smb:::op-WriteX-start
smb:::op-WriteX-done
smb2:::op-Write-start
smb2:::op-Write-done
```

For descriptions of the arguments to the DTrace probes, see SMB DTrace Arguments. For how to use all the probes in a script, see SMB DTrace Examples.

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);
}
```

The following example DTrace script shows how to trace all SMB2 requests.

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

#pragma D option quiet

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

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

```
}

smb2:::op-Negotiate-start,
smb2:::op-SessionSetup-start,
smb2:::op-Logoff-start,
smb2:::op-TreeConnect-start,
smb2:::op-TreeDisconnect-start,
smb2:::op-Create-start,
smb2:::op-Close-start,
smb2:::op-Flush-start,
smb2:::op-Read-start,
smb2:::op-Write-start,
smb2:::op-Lock-start,
smb2:::op-Ioctl-start,
smb2:::op-Cancel-start,
smb2:::op-Echo-start,
smb2:::op-QueryDirectory-start,
smb2:::op-ChangeNotify-start,
smb2:::op-QueryInfo-start,
smb2:::op-SetInfo-start,
smb2:::op-OplockBreak-start
{
        self->thread = curthread;
        self->start = timestamp;
}

smb2:::op-Negotiate-done,
smb2:::op-SessionSetup-done,
smb2:::op-Logoff-done,
smb2:::op-TreeConnect-done,
smb2:::op-TreeDisconnect-done,
smb2:::op-Create-done,
smb2:::op-Close-done,
smb2:::op-Flush-done,
smb2:::op-Read-done,
smb2:::op-Write-done,
smb2:::op-Lock-done,
smb2:::op-Ioctl-done,
smb2:::op-Cancel-done,
smb2:::op-Echo-done,
smb2:::op-QueryDirectory-done,
smb2:::op-ChangeNotify-done,
smb2:::op-QueryInfo-done,
smb2:::op-SetInfo-done,
smb2:::op-OplockBreak-done
/self->thread == curthread/
{
        printf("%39s/%-17d %-31s %8d 0x%08x %17d %9d %17d 0x%08x\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_tid,
           args[1]->soi_asyncid,
               args[1]->soi_flags);
}
```

The following example DTrace script how to trace SMB2 reads and writes.

```
#!/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");
}

smb2:::op-Read-start,
smb2:::op-Write-start
{
        self->thread = curthread;
        self->start = timestamp;
}

/*
 * The following action is executed if the field 'soi_curpath' is undefined (or
 * NULL).
 */
smb2:::op-Read-done,
smb2:::op-Write-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).
```

```
*/
smb2:::op-Read-done,
smb2:::op-Write-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);
}
```

B

Commonly Used SMB File Sharing Commands

Managing SMB File Sharing

The following list shows the basic command syntax for performing some common SMB file sharing tasks in the Oracle Solaris 11.4 release. For more information, see the smbadm(8), smb(5), share(8), and mount(8) man pages.

- Enable SMB sharing for ZFS file system on the dataset

  ```
  $ zfs set share.smb=on pool/dataset
  ```

- Create an SMB share having no default property value

  ```
  $ zfs share -o share.smb=on pool/dataset%share-name
  ```

- Find share to mount from the server

  ```
  $ smbadm show-shares server
  ```

- Mount share on a directory

  ```
  $ mount -F smbfs //server/share mount-point
  ```

- Remove an SMB Share

  ```
  $ zfs destroy pool/dataset%share-name
  ```

- List all SMB mounts

  ```
  $ mount -v | grep 'type smbfs'
  ```

- View all the SMB environment property values

  ```
  $ sharectl get smb
  ```

- Create an SMB group

  ```
  $ smbadm create-group group-name
  ```

- Add a member to an SMB group

  ```
  $ smbadm add-member -m member-name group-name
  ```

- Remove member from an SMB group

  ```
  $ smbadm remove-member -m member-name group-name
  ```

- Modify an SMB group property

  ```
  $ smbadm set-group -p property=value group-name
  ```

- Store SMB persistent passwords

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

- Remove SMB persistent passwords

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

Glossary

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.

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.

SMB Common Internet File System

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.

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.

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 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.

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 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.

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 the type `disk` because the shares are directories. A share of type `pipe` represents a device, such as an IPC share.

Server Message Block (SMB)

A protocol that enables clients to access files and to request services of a server on the network.

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 a security and policy database. In a Windows workgroup, computers cooperate through the use of a common workgroup name but this peer-to-peer model has no formal membership mechanism.

Index

L

M

N

O

P

R

S

T

U

V

W