Abstract

This technical report covers how to configure Lightweight Directory Access Protocol (LDAP) identity management for multiprotocol NAS in NetApp® ONTAP® based systems. This document complements, and can be considered a replacement for, TR-4073: Secure Unified Authentication. For information about name services best practices, see TR-4668: Name Services Best Practices Guide.
# Table of Contents

1 Overview .................................................................................................................. 5
   1.1 What Is LDAP? .................................................................................................. 5
   1.2 What Is Multiprotocol NAS? ........................................................................... 6
   1.3 What Is CIFS/SMB? ......................................................................................... 6
   1.4 What Is NFS? .................................................................................................. 7

2 Authentication in ONTAP ......................................................................................... 7
   2.1 Homogenous Versus Heterogenous NAS Environments ............................... 7
   2.2 Why Should You Use Multiprotocol NAS? .................................................... 7
   2.3 Common Challenges in Multiprotocol Access .............................................. 8

3 LDAP Components and Considerations ................................................................. 8
   3.1 LDAP Basics ................................................................................................... 8
   3.2 How ONTAP Processes Authentication Requests by Using LDAP ............... 9
   3.3 LDAP Schemas .............................................................................................. 14
   3.4 LDAP Search Distinguished Names and Scopes ............................................ 17
   3.5 Centrify Integration and Considerations ....................................................... 22
   3.6 LDAP Referrals (Chase Referrals) ................................................................. 23

4 Configuration .......................................................................................................... 24
   4.1 LDAP Environment Information—Preconfiguration ........................................ 24
   4.2 LDAP Environment Configuration ................................................................. 30
   4.3 LDAP That Uses Active Directory for UNIX Identity Management .............. 37
   4.4 Configure Secure LDAP ................................................................................ 57
   4.5 Use of LDAP to Host Netgroups ..................................................................... 65
   4.6 Configure External LDAP Clients ................................................................. 77
   4.7 LDAP Authentication for Cluster Administration ........................................... 83

5 Common Issues and Troubleshooting Steps ......................................................... 89
   5.1 Optimize LDAP Searches .............................................................................. 89
   5.2 Points of Failure ............................................................................................ 90
   5.3 Troubleshooting Tools ................................................................................... 92
   5.4 What Information to Collect Before You Contact NetApp Support .............. 116

6 Best Practices ......................................................................................................... 117
   6.1 LDAP Server Best Practices ......................................................................... 117
   6.2 DNS Server Best Practices .......................................................................... 118
   6.3 Cache Management Best Practices .............................................................. 118
1 Overview

This technical report covers Lightweight Directory Access Protocol (LDAP) configuration as a method for UNIX identity management and name mapping for multiprotocol NAS access on NetApp® storage systems that run NetApp ONTAP® software. The use of LDAP as a centralized name service provides scale, consistency, and ease of management in multiprotocol NAS environments. Multiprotocol NAS access enables enterprise and scale-out storage systems to provide access to clients that are running both Linux and Windows-based operating systems through the NFS and CIFS/SMB protocols, respectively.

This document focuses on:

- ONTAP 9.7 and later
- CentOS/Red Hat Enterprise Linux (RHEL) 7 and later
- FreeIPA 4.8 and later
- Microsoft Active Directory LDAP on Windows 2012 and later

For other client and LDAP server configurations, contact the product vendor. Most earlier ONTAP versions apply to this TR as well, but they might be missing some of the features or commands that are mentioned in this document.

1.1 What Is LDAP?

LDAP is a standard directory access protocol that was developed by an international committee called the Internet Engineering Task Force (IETF). LDAP is intended to provide a general-purpose, network-based directory service that you can use across heterogeneous platforms to locate network objects. LDAPv3 is the standard currently implemented version.

LDAP models define how to communicate with the LDAP directory store, how to find an object in the directory, how to describe the objects in the store, and the security that is used to access the directory. LDAP allows customization and extension of the objects that are described in the store. Therefore, you can use an LDAP store to store many types of diverse information. Many of the initial LDAP deployments focused on the use of LDAP as a directory store for applications such as email and web applications and to store employee information. During the past several years, LDAP has been gaining acceptance as a directory store for information that is used in network-based authentication and authorization. Many companies are replacing the Network Information Service (NIS) with LDAP as a network directory store.

LDAP for UNIX identity management can be served through Active Directory or any RFC 2307–compliant LDAP provider. ONTAP software supports a multitude of LDAP servers and can claim support for any LDAP server that adheres to the standards that IETF has laid out.

You can also use LDAP for identity management for cluster administration logins, but this document does not cover the scope of that use. For LDAP use with cluster logins, see the product documentation.

Microsoft Active Directory as a UNIX LDAP Server

Microsoft implemented LDAPv3 as a directory store starting in Windows 2000/2003 Active Directory. The Microsoft LDAP implementation is standards based, so you can use Microsoft Active Directory LDAP to store UNIX user and group information. With this capability, you can unify the directory service and the directory store of networks based on both Windows and UNIX. Before Windows 2008 R2, native Active Directory LDAP did not contain the definitions of attributes needed to hold information that is necessary for UNIX authentication and authorization. Therefore, the Microsoft Active Directory schema needed to be extended with the necessary objects to hold this information in those versions. Windows 2008 R2 and later offer UNIX schema extensions in Active Directory by default and require no schema modifications to implement them.
What Does LDAP Store?

LDAP can store the following information that is used in multiprotocol NAS access:

- User names
- Group names
- Numeric user IDs (UIDs) and group IDs (GIDs)
- Home directories
- Login shell
- Netgroups, DNS names, and IP addresses
- Group membership

How Does ONTAP Interact with LDAP?

ONTAP can use LDAP in one of two ways:

- Querying for user names, numeric IDs, groups, group memberships, netgroups, name mappings, and so on, for NAS protocol operations
- Interacting with Privileged Access Management (PAM) for cluster and ONTAP System Manager logins for cluster administration.

This document covers mostly LDAP for use with NAS protocol operations, but section 4.7, "LDAP Authentication for Cluster Administration", covers the use of LDAP servers to host users and groups in cluster logins/authentication for other access.

1.2 What Is Multiprotocol NAS?

Multiprotocol NAS is essentially what the name suggests: unified NAS access through multiple NAS protocols. The use of multiprotocol NAS on NetApp storage systems enables users on all operating systems to seamlessly access the same datasets, regardless of the type of protocol that is used. The protocols that are involved in multiprotocol environments are CIFS/SMB and NFS.

Is Multiprotocol NAS Also Called Mixed Mode?

A common misconception is that multiprotocol NAS is also called mixed mode. This belief creates confusion when implementing a NetApp storage system that runs NAS, because the concept of mixed security style also exists. Mixed security style is covered later in this document, in a subsection of section 3 titled “Security Styles.”

1.3 What Is CIFS/SMB?

CIFS (Common Internet File System)/Server Message Block (SMB) is the way that users share files across Ethernet-based networks primarily on operating systems that run Microsoft Windows. CIFS is the native file-sharing protocol that was introduced in Windows 2000, and it uses SMB as the underlying protocol for communication between clients and servers in modern operating systems.

CIFS/SMB is also used on other non-Windows operating systems such as Apple, Linux, and Oracle Solaris through third-party implementations, such as Samba. Support of CIFS/SMB on non-Windows operating systems on NetApp storage systems varies and can be found on the Interoperability Matrix Tool (IMT). For more information about CIFS/SMB in ONTAP, see TR-4191: Best Practices Guide for ONTAP 8.2.x Windows File Services.

Note: Although CIFS and SMB are different in many ways, this document uses the terms interchangeably.
1.4 What Is NFS?

NFS (Network File System) is the way that users share files across Ethernet-based networks primarily on operating systems that run Linux, Oracle Solaris, UNIX, HP-UX, and so on. NFS follows a series of standards defined by the IETF through documents called Request for Comments (RFC). These standards are followed by all major NFS client and server vendors who intend to deliver enterprise-level NFS access. NFS depends on a series of underlying messages, and those underlying messages depend on the version of NFS that is being used. For more information about NFS in ONTAP, see TR-4067: NFS Best Practices and Implementation Guide.

2 Authentication in ONTAP

NetApp ONTAP is built on a UNIX-based operating system, so native NFS environments generally have little trouble gaining access, because the underlying methodologies are the same, particularly when volumes and qtrees use UNIX security styles. In this report, a subsection of section 3.2, “Security Styles,” presents more information.

However, when volume and qtree security styles are NTFS, to gain access, NFS clients must perform a user mapping to a valid Windows user. This step is necessary because NTFS access control lists (ACLs) are not understood by NFS clients. Therefore, the storage system must act as an arbitrator for the client to determine whether the user can authenticate and has access through the NTFS ACL.

CIFS/SMB environments run into challenges because clients that use those protocols must pass a generic Windows -> UNIX user authentication before they gain access to a system, even when volume security styles are NTFS. Also, user mapping to ensure that file owners are represented accurately in mixed-protocol environments can add another wrinkle for storage administrators.

2.1 Homogenous Versus Heterogenous NAS Environments

Some sites have pure Windows or pure UNIX environments in which all data is accessed by using only one of the following:

- CIFS/SMB and NTFS file security
- NFS and UNIX file security (mode bits or NFSv4.x ACLs)

However, many sites must enable datasets to be accessed from both Windows and UNIX clients. For these environments, ONTAP has native multiprotocol NAS support. After the user is authenticated on the network and has both appropriate share or export permissions and the necessary file-level permissions, the user can access data from UNIX hosts by using NFS or from Windows hosts by using CIFS/SMB. The use of multiprotocol NAS access does not require the use of mixed-security-style (sometimes referred to as “mixed-mode”) volumes and qtrees, even though it is available as an option.

2.2 Why Should You Use Multiprotocol NAS?

By using multiprotocol NAS with ONTAP, you gain several distinct advantages. When clients can use different NAS protocols to seamlessly access datasets simultaneously, you can achieve the following benefits:

- Reduced overall storage administrator management tasks
- Requirement for only a single copy of data to be stored for NAS access from multiple clients
- Protocol-agnostic NAS that enables storage administrators to control the style of ACL and the access control that is presented to end users
- Centralized identity management operations in a NAS environment
ONTAP has provided enterprise-class multiprotocol NAS access for over 25 years and counting. With the advent of scale-out ONTAP clusters and NetApp ONTAP FlexGroup volumes, storage administrators have even more flexibility with multiprotocol NAS environments.

**Multiprotocol NAS Use Cases**

The most common ways to use multiprotocol NAS include, but are not limited to:

- Home directories
- Source code repositories
- Research and engineering shares
- Image repositories
- Audio and video editing and rendering

**2.3 Common Challenges in Multiprotocol Access**

Many organizations want to use multiprotocol NAS access for its flexibility. However, there is a perception of difficulty in multiprotocol NAS that creates a specific set of challenges that are unique to the concept of sharing across protocols. This perception is grounded in reality, but only if the underlying infrastructure has not been prepared for multiprotocol NAS access. For example, standing up an LDAP server for identity management needs can greatly simplify multiprotocol NAS environments.

These challenges include, but are not limited to:

- Requirement of knowledge across multiple protocols, operating systems, and storage systems
- Working knowledge of name service servers, such as DNS, LDAP, and NIS
- External factors such as:
  - Dealing with multiple departments and IT groups (for example, a Windows group and a UNIX group)
  - Company acquisitions
  - Domain consolidation
  - Reorganizations
  - Many moving parts

By using a viable name service such as LDAP for UNIX identity management, you can greatly simplify the way that ONTAP can serve multiprotocol NAS. When you use LDAP along with Active Directory, it provides a centralized name service for both NFS/UNIX and SMB/Windows identities, which further simplifies operations.

**3 LDAP Components and Considerations**

NetApp ONTAP presents storage management by way of a series of concepts known as storage virtual machines (SVMs). These components act as individual storage silos for multitenant storage operations. As such, LDAP configuration and operations are performed at the SVM level, which means that each SVM can use its own LDAP client configuration and schema as desired. ONTAP and its SVMs act as LDAP clients to LDAP servers, just like any other NAS client would. To maintain consistency of users, groups, and name mapping rules without having to configure them at multiple sources, ONTAP and NAS clients can all share the same LDAP server sources.

**3.1 LDAP Basics**

The following section covers some high-level basics for LDAP and how it works:
• By default, LDAP operates on TCP port 389 for normal traffic and operates on port 636 for secure LDAP that uses Secure Sockets Layer (SSL). You can modify LDAP ports for security purposes. Active Directory LDAP can also serve LDAP traffic on the global catalog port 3268. The secure global catalog port (3269) is currently unsupported in ONTAP.

• LDAP information is stored in flat files in an LDAP server and is organized by way of an LDAP schema. You should configure LDAP clients in a way that coordinates their requests and lookups with the schema on the LDAP server.

• LDAP clients initiate queries by way of an LDAP bind, which is essentially a login to the LDAP server. The LDAP bind configuration on the clients is configured to use the security mechanism that is defined by the LDAP server. At a minimum, LDAP servers allow anonymous binds, which are the least secure. Most LDAP servers rely on some level of security mechanism for binds. In some cases, they are simple user name and password exchanges. In other cases, binds are secured through SSL or Kerberos/Generic Security Service API (GSSAPI) for encrypted communication. ONTAP supports all these methods for LDAP binds. In addition, ONTAP allows LDAP binds to Windows Active Directory through the CIFS/SMB machine account.

• User and group information that is stored in LDAP is queried by clients by using standard LDAP search requests as defined in RFC 2307. In addition, newer mechanisms, such as RFC 2307bis, allow more streamlined user and group lookups. You can also configure these lookups and replies to be protected by encryption by way of LDAP client and server configuration.

• LDAP servers can store user and group information as well as netgroup information for use with NFS export rule configuration.

• If the initial requests are not found on the initial server, LDAP servers can refer requests to other LDAP servers by way of chase referrals. Starting in version 9.5, ONTAP supports chase referrals; for more information, see section 3.6, "LDAP Referrals (Chase Referrals)."

• You can configure LDAP queries in ONTAP to time out after a set number of seconds. By default, queries time out after 3 seconds.

• To speed up queries and to avoid crawling large LDAP schemas, you can filter down LDAP queries based on specified locations and distinguished names (DNs).

• You can define LDAP servers in the LDAP client configuration in ONTAP through IP address or host name, or you can simply use DNS service (SRV) records to find the associated LDAP servers.

3.2 How ONTAP Processes Authentication Requests by Using LDAP

When a user attempts to authenticate into an ONTAP NAS share, ONTAP tries to figure out the identity of the user to establish whether that user can access what they are requesting. Name lookups are also vital to name mappings in ONTAP.

An initial authentication occurs, which is a verification of the user to gather numeric UID, GID, and group membership and to determine whether the user who is requesting access actually exists in the system. Where ONTAP gathers this information depends on how you configure the name service switch (ns-switch) for the SVM. Valid name service sources for user and group information are local files (passwd and group), NIS, and LDAP. ONTAP queries the specified name service sources in the order that they are listed.

Configure the name service switch through the ns-switch command set.

```
cluster::> ns-switch ?
(vserver services name-service ns-switch)
create                      Create a new Name Service Switch table entry
delete                      Remove a Name Service Switch table entry
modify                      Change a Name Service Switch table entry
show                        Display Name Service Switch configuration

cluster::> ns-switch show -vserver DEMO
(vserver services name-service ns-switch show)
Source
```
<table>
<thead>
<tr>
<th>Vserver</th>
<th>Database</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>hosts</td>
<td>dns,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>files</td>
</tr>
<tr>
<td>DEMO</td>
<td>group</td>
<td>ldap,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>files</td>
</tr>
<tr>
<td>DEMO</td>
<td>passwd</td>
<td>ldap,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>files</td>
</tr>
<tr>
<td>DEMO</td>
<td>netgroup</td>
<td>ldap,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>files</td>
</tr>
<tr>
<td>DEMO</td>
<td>namemap</td>
<td>ldap,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>files</td>
</tr>
</tbody>
</table>

If you specify LDAP, then when a user lookup is needed, ONTAP uses logical interfaces (LIFs) in the SVM that have access to the LDAP servers. These LIFs can be either data LIFs or SVM management LIFs. If no LIFs in the SVM can reach LDAP servers, name service requests fail.

When ONTAP has determined that LDAP will be used for the name lookup, the following process occurs:

1. ONTAP first checks the connection cache to see whether connections to LDAP servers are already established. If no connections are cached, ONTAP uses the LDAP client configuration to make a connection attempt to the servers that are specified in the configuration. If host names are specified, DNS lookups are performed. If Active Directory domains are used, DNS SRV record lookups are performed.

2. If the TCP connection over the defined LDAP service port is successful, then ONTAP attempts to “bind” (log in) to the LDAP server by using the defined credentials in the client configuration.

3. If the bind is successful, then ONTAP uses the client schema that is defined in the LDAP client to make an LDAP search query to the LDAP server. The following information is passed to the server in the query:
   - Base/user DN (to narrow search scopes)
   - Search scope type (subtree, base, onelevel)
   - Object class (to search only for objects in that class)
   - UID/user name
   - Requested attributes (uid, uidNumber, gidNumber, unixUserPassword, name, unixHomeDirectory, loginShell)

4. If the user is not found in the first request and chase referrals are enabled, ONTAP tries other LDAP servers. If the user is not found and multiple DNs are defined in the client configuration, ONTAP tries other DNs. If the user is not found in any scenario, an error is returned and ONTAP tries the next available ns-switch source. If no user is found there, the request fails and access is denied.

5. If the request is successful, ONTAP stores the user attributes in a cache for future use.

The flow of operations is important to remember if failures occur, because it can help narrow down why and how LDAP requests might not be working. For more information about LDAP lookup failures, see section 5, “Common Issues and Troubleshooting Steps.”

**Security Styles**

In ONTAP, permissions are managed by how the security style is configured for a volume, qtree, file, or folder. ONTAP storage administrators can manage security styles from the cluster for volumes and qtrees only. File and folder security styles are set based on where the file or folder is written. For example, if a file is written or copied to a volume with the NTFS security style, then the file also is assigned the NTFS security style. Flipping the security style of a volume or qtree does not change the security styles or ACLs of existing files or folders. In ONTAP, each volume can have its own independent security style, and each qtree can also have its own security style. This approach allows flexibility in management of user data, especially in-home directory scenarios, where some users might want to manage permissions from Windows clients and other users might want to manage permissions from Linux clients.
Security styles are used to help maintain a consistent set of permissions, especially when you use both the NFS and SMB protocols on the same sets of data in a cluster. Security styles are really just a way of saying “permission styles”—how do you want to manage the ACLs for your data?

ONTAP has three security styles for use with multiprotocol NAS:

- **NTFS.** This style uses standard Windows permission models and logic. The same rules that apply to Windows permissions apply to NTFS security style volumes and qtrees. Only Windows clients can change permissions and owners on volumes and qtrees by using NTFS security styles. NFS clients that try to change permissions fail. You might or might not see an error, depending on the way that you configure the NFS server and export policies for NFS.

  You can also manage NTFS permissions from the ONTAP cluster CLI by using Storage-Level Access Guard or the `vserver security file-directory` command sets, all the way down to the file and folder level.

- **UNIX.** This security style uses UNIX-style permission structures. It includes mode bits (for example, `chmod 755`) for read/write/execute with owner/group/others, as well as NFSv4.x ACLs for more granular ACL management for users and groups. It does not include POSIX ACLs, which are not supported in ONTAP. UNIX security style volumes and qtrees manage permissions by using only NFS clients. Windows clients can view mode bit permissions depending on the CIFS server settings, but NFSv4.x ACL viewing is possible only over SMB 1.0, which is deprecated. SMB 2.0 and later versions can’t parse the NFSv4.x ACL information properly. Windows clients cannot change UNIX permissions or owners.

  You can set UNIX mode bit permissions from the cluster CLI, but only on top-level volumes and qtrees. File permissions are set from clients. You can set file owners from the ONTAP CLI only when you use NetApp FlexClone® volume technology. You can set NFSv4.x ACLs only from clients that use NFSv4.x, and only if you have enabled NFSv4.x ACL support on the NFS server in ONTAP. This ACL management limitation includes security and audit style ACLs.

- **Mixed.** This security style is always either NTFS or UNIX at an effective level. However, unlike the other security styles, mixed allows the effective style to change based on the style of ACL that last changed permissions on the file or folder. This style is useful when clients must be able to change permissions from any protocol at any given moment, such as applications that might write from NFS clients and then need to modify permissions later from Windows clients. The nature of mixed security style and its ability to change can create complexity in multiprotocol NAS environments, especially when name mapping or name services are not configured properly. As such, NetApp does not recommend mixed security styles unless you need a specific use case.

For information about how to view security styles and permissions from the ONTAP CLI, see the subsection titled “vserver security file-directory show.”

### Name Mapping in ONTAP

Name mapping is the way that ONTAP (on a Linux/UNIX-based operating system) handles multiprotocol NAS access between NFS and CIFS/SMB protocols. Because volumes are always either UNIX or NTFS security styles, name mapping is a way to confirm that the appropriate ACLs that you set on data objects are being enforced.

By default, ONTAP maps users with identical names in Windows and UNIX without the need for any administrator intervention by way of name mapping rules. For example, a UNIX user named “techontap” maps implicitly to a Windows user named “techontap” without needing any special name mappings.

If a user has a different name in Windows from what they have in UNIX (or vice versa), then ONTAP needs more information to properly map users. This information can come from one of three places:

- Name service mapping (such as LDAP)
- Name mapping rules that are set at the SVM level
- Default UNIX or Windows user (set at the NFS or CIFS server level)
If no valid name mapping exists for a user, the authentication request fails and a client sees access or permission denied when attempting to access NAS data that is hosted in ONTAP.

If a valid name mapping exists but maps a UNIX or Windows user to the wrong Windows or UNIX user, a range of outcomes can result. Outcomes range from incorrect access to files and folders to incorrect owners being set on files and folders (such as CIFS files showing up as nobody or 65534 on NFS exports).

Name Mapping Rules

In ONTAP, you can configure name mapping rules for individually specified users or even for host names and clients. Name mapping rules cannot be used for group-to-user mappings, however.

You can use name mapping rules for three different directional purposes, which are specified by the option -direction:

- **win-unix** is used for mapping Windows user names to UNIX user names. You can specify the Windows Active Directory domain in the user name by using `DOMAIN\user` format. These name mapping rules help ONTAP determine the appropriate UNIX user to use when determining permissions and access rights on UNIX security style objects.

- **unix-win** is used for mapping UNIX names to Window names. UNIX names are specified by user name, because they are needed to map into Windows. ONTAP must be able to translate incoming UNIX numeric UIDs to names through name service servers or local files. Otherwise, if no UNIX user name can map to the UID, then the mapping request uses the numeric UID to look for a Windows user of `DOMAIN/numericID`. These name mapping rules help ONTAP determine the appropriate Windows user to use when determining permissions and access rights on NTFS security style objects.

- **krb-unix** is used for NFS Kerberos service principal name (SPN) mappings. By default, NFS Kerberos SPNs attempt to map by using either the service name (such as the `nfs` portion of `nfs/host@DOMAIN.COM`) or the `user/machine account` portion of `NAME@DOMAIN.COM` in Kerberos requests. In some cases, storage administrators might want to control the name mapping of those SPNs rather than letting ONTAP map 1:1. TR-4616 covers these scenarios in more detail.

Regular Expressions

ONTAP allows the use of standard regular expression (regex) values in name mapping rules. They can provide global replacement and wildcard functionality for name mapping rules when UNIX and Windows names are not 1:1 but are close enough to take care of with a simple regex value.

For example, if all Windows names follow the format `first.last` for names and the UNIX environment uses `first_last`, then you could use a regex to map all dots to underscores in user names. That regex would look like the following:

```
cluster::*> vserver name-mapping show -vserver DEMO -direction win-unix
Vserver: DEMO
Direction: win-unix
Position Hostname IP Address/Mask
------- ---------------- -------------
1 - - Pattern: (.+)\.(.+).(.)
Replacement: \2_\3
```

Without the preceding rule, user names like `user.name` would map to the default user:

```
cluster::*> diag secd name-mapping show -node ontap9-tme-8040-01 -vserver DEMO -direction win-unix -name NTAP\user.name
'NTAP\user.name' maps to 'pcuser'
```

With that rule, `DOMAIN\user.name` maps to the UNIX user `user_name`.
Mapping the Administrator to the Root (and Vice Versa)

ONTAP provides options for mapping all administrative users to the root and for mapping the root to the Windows administrator. The purpose is to treat all administrative users the same across multiple NAS protocols regardless of name mapping rules.

For example, if a Windows user named prof1 has a valid UNIX user name of prof1, then the default name mapping is NTAP\prof1 == prof1.

If Windows user prof1 is a member of the domain or local administrators group, then enabling the CIFS/SMB option -is-admin-users-mapped-to-root-enabled maps that user to the root.

The following example shows output of the show-creds command for the NTAP\prof1 Windows user. Note the name mapping and the Windows group membership (in **bold**):

<table>
<thead>
<tr>
<th>Cluster:</th>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>cifs option show -vserver DEMO -fields is-admin-users-mapped-to-root-enabled</td>
<td>vserver is-admin-users-mapped-to-root-enabled</td>
<td>DEMO true</td>
</tr>
<tr>
<td>vserver services access-check authentication show-creds -vserver DEMO -win-name NTAP\prof1</td>
<td>UNIX UID: prof1 &lt;&gt; Windows User: NTAP\prof1 (Windows Domain User)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GID: ProfGroup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Supplementary GIDs:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ProfGroup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>group1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>group2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>group3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sharedgroup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Primary Group SID: NTAP\DomainUsers (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows Membership:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\group2 (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\DomainUsers (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\sharedgroup (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\group1 (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\group3 (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTAP\ProfGroup (Windows Domain group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Service asserted identity (Windows Well known group)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BUILTIN\Backup Operators (Windows Alias)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BUILTIN\Users (Windows Alias)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>User is also a member of Everyone, Authenticated Users, and Network Users</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Privileges (0x2086):</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SeBackupPrivilege</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SeRestorePrivilege</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SeChangeNotifyPrivilege</td>
<td></td>
</tr>
</tbody>
</table>

When that user is added to the local administrators group for the SVM, it now gets mapped to the root.

```
class:>> local-group add-members -vserver DEMO -group-name BUILTIN\administrators -member-names NTAP\prof1 (vserver cifs users-and-groups local-group add-members)
```

```
class:>> local-group show-members -vserver DEMO -group-name administrators (vserver cifs users-and-groups local-group show-members)
```

Vserver: DEMO
### Use of LDAP for Name Mapping

LDAP can be a name mapping resource, provided that the LDAP schema on the server has been populated properly and that the ONTAP SVM’s LDAP client schema reflects the assigned LDAP attributes that are used for name mapping. For example, to map UNIX users to corresponding Windows user names that do not match 1:1, you can use a combination of `uid` (for UNIX user names) and `sAMAccountName` (for Windows user names) in the LDAP client configuration. The attributes that are used depend on what is found during the LDAP preconfiguration steps, which are covered in section 4, “Configuration.”

#### 3.3 LDAP Schemas

LDAP schemas are how LDAP servers organize and collect information. LDAP server schemas generally follow the same standards, but different LDAP server providers might have variations on how schemas are presented. ONTAP has built-in read-only schemas that are available to administrators. You can use

---

When a user is mapped to the root and writes files to a NAS share, then the ownership reflects that name mapping. In the following example, `prof1` wrote a file from SMB called `mappedroot.txt` and the owner became `root`:

```
# su prof1
sh-4.2$ cd /profgroup
sh-4.2$ ls -la
total 8
drwxrwxrwx 2 root root 4096 Feb 21 14:59 .
drwxrwxrwx 11 root root 4096 Feb  3 09:34 ..
-rwxrwxrwx 1 root ProfGroup 0 Feb 21 14:59 mappedroot.txt
-rw-r--r-- 1 prof1 ProfGroup 0 Aug 30 10:30 newfile1
-rw-r--r-- 1 prof1 ProfGroup 0 Aug 30 10:30 newfile2
```

---

---
these schemas to configure LDAP, or you can copy them to read/writable schemas to allow modification of the schema attributes for LDAP servers that do not contain the same attributes as any of the default schemas. LDAP schemas must exist before you configure an LDAP client.

In LDAP queries by the cluster, schemas are used to speed up name lookups because they enable the use of specific attributes to find information about a user, such as the UID. The schema attributes must exist in the LDAP server for the cluster to be able to find the entry. Otherwise, LDAP queries might return no data and authentication requests might fail.

For example, if a UID number (such as \texttt{root=0}) must be queried by the cluster and the cluster is configured to use AD-IDMU schema styles, then the schema attribute \texttt{RFC 2307 uidNumber Attribute} is used. The default schema for AD-IDMU uses the attribute \texttt{uidNumber} for that query.

If the LDAP server uses a different schema attribute for that information, then the query can’t find that information.

For ease of use and configuration, ONTAP provides several default, read-only schema templates. These schema templates generally correspond with specific LDAP servers. Table 2 shows the LDAP schema templates that ONTAP provides and the corresponding LDAP server for each. In other words, if you use a specific type of LDAP server, then the schema that is listed with it should work for that server in most cases.

**RFC 2307bis**

\texttt{RFC 2307} is the Request for Comments memo titled “An Approach for Using LDAP as a Network Information Service.” \texttt{RFC 2307bis} is an extension of RFC 2307 and adds support for \texttt{posixGroup}, which enables dynamic lookups for auxiliary groups by using the \texttt{uniqueMember} attribute, rather than by using the \texttt{memberUid} attribute in the LDAP schema. Instead of using just the name of the user, this attribute contains the full distinguished name (DN) of another object in the LDAP database. Therefore, groups can have other groups as members, which allows nesting of groups. Support for RFC 2307bis also adds support for the object class \texttt{groupOfUniqueNames}.

This RFC extension fits very nicely into how Microsoft Active Directory manages users and groups through the usual management tools. For example, without RFC 2307bis support, if you want to ensure that a user’s supplemental groups are populated in LDAP queries, you must populate the \texttt{memberUid} field in the necessary groups with the users that live in that group. You perform this step through the classic UNIX Attributes tab, which is available in older versions of Active Directory (deprecated in Windows 2012 and later), or through the Attributes Editor tab or Windows PowerShell in newer versions of Windows. That task is an additional step to the normal Windows group management method of simply adding users as members of groups.

With RFC 2307bis, when you add a Windows user to a group, as long as that group has a valid numeric GID, LDAP lookups pull the necessary supplemental group information from the usual Windows attribute and find the numeric GIDs automatically.

As such, \textbf{NetApp highly recommends} that you enable RFC 2307bis when you configure LDAP clients in ONTAP. You enable it through the schema creation, and it is available by default with the \texttt{MS-AD-BIS} schema that ONTAP provides.

**Group Membership and Supplemental Groups**

You can use LDAP to control group membership for users and to return supplemental groups for a user. This behavior is controlled through schema attributes.

**Primary GID**

For ONTAP to be able to search properly, LDAP users must always have a primary GID defined. The user’s primary GID is defined by the schema attribute \texttt{-gid-number-attribute}. In most cases, that
attribute is gidNumber. In some cases, it might be gid or primaryGid. Be sure to check with your LDAP administrator for the proper attribute.

If a user has no primary numeric UNIX GID, then ONTAP fails the request, even if the user has a valid UNIX UID and numeric.

```
cluster::*> getxxbyyyy getpwbyname -node node1 -vserver DEMO -username test -show-source true
-use-cache false -show-granular-err true
(vserver services name-service getxxbyyyy getpwbyname)
NIS:
Error code:    NS_ERROR_NONE
Error message: No error
LDAP:
Error code:    NS_ERROR_NOT_FOUND
Error message: Entry not found
DNS:
Error code:    NS_ERROR_NONE
Error message: No error
FILES:
Error code:    NS_ERROR_NOT_FOUND
Error message: Entry not found
Deterministic Result: Authoritative Error
```

Error: command failed: Failed to resolve test. Reason: Entry not found for "username: test".

**Secondary, Supplemental, and Auxiliary GIDs**

Secondary, supplemental, and auxiliary groups are groups that a user is a member of outside of their primary GID. They are defined in two ways in LDAP:

- memberUid
- RFC 2307bis

LDAP can query group membership based on the LDAP schema configuration in use. For example, if RFC 2307bis is enabled, then LDAP looks up based on the RFC 2307bis attributes that are defined in the schema; see the "RFC 2307bis" section of this document. If supplemental groups do not appear as expected for a user, then the likely issue is that the LDAP client schema is misconfigured. For more information, see section 5, "Common Issues and Troubleshooting Steps."

**Increasing the Number of Allowed Supplemental and Auxiliary GIDs**

Remote Procedure Call (RPC) has a specific limitation for the maximum number of auxiliary GIDs that can be honored in a single NFS request. The maximum for AUTH_SYS/AUTH_UNIX is 16, and for AUTH_GSS (Kerberos), it is 32. This protocol limitation affects many NFS servers, not just ONTAP. To work around this NFS limitation in ONTAP, use the following NFS server options:

```
auth-sys-extended-groups
extended-groups-limit
```

Also, the LDAP client schema has an option to increase the supported groups for use with RFC 2307bis, which defaults to 256 and can increase to 1,024.

```
-maximum-groups-rfc2307bis
```

**How It Works**

The options to extend the group limitation work the same way that the manage-gids option for other NFS servers works. Basically, rather than dumping the entire list of auxiliary GIDs that a user belongs to, the option performs a lookup for the GID on the file or folder and returns that value instead (Figure 1).
Any GID past the limit of 16 is dropped by the protocol. With the extended GID option in ONTAP, when a new NFS request comes in, ONTAP requests information about the user's group membership. You can use extended GIDs with external name services, or if the users and groups are configured properly, you can use them locally on the cluster. If you use local files, make sure that a local UNIX user is a member of multiple groups with the `unix-group adduser(s)` command.

**Performance Impact of Extended GIDs**

Extended groups have a minimal performance penalty, generally in the low single-digit percentages. Higher metadata NFS workloads likely have more impact, particularly on the system’s caches. Performance can also be affected by the speed and workload of the name service servers. Overloaded name service servers are slower to respond, causing delays in prefetching the GID.

**Considerations for Extended GIDs with Active Directory LDAP**

By default, in Microsoft Active Directory LDAP servers, the `MaxPageSize` attribute is set to a default of 1,000. That setting means that groups beyond 1,000 would be truncated in LDAP queries. To enable full support with the 1,024 value for extended groups, the `MaxPageSize` attribute must be modified to reflect the 1,024 value. For information about how to change that value, see the Microsoft TechNet article [How to View and Set LDAP Policy in Active Directory by Using Ntdsutil.exe](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/hh831778(v=ws.10)).

If you have concerns about modifying this value, contact Microsoft support and review the TechNet library article [MaxPageSize Is Set Too High](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/hh831778(v=ws.10)).

### 3.4 LDAP Search Distinguished Names and Scopes

The following section covers LDAP schema architecture and how searches are conducted through the search using distinguished names (DNs) and scopes.

**Distinguished Names**

A **DN** is a sequence of relative DNs (RDNs), separated by commas. In LDAP, it is essentially a folder structure that specifies the locality of objects such as users, groups, machine accounts, and netgroups.
In Active Directory, for instance, the domain itself can be represented as a DN. A domain of domain.netapp.com becomes a DN of dc=domain,dc=netapp,dc=com.

Table 1 shows what common RDN types are used in DNs.

<table>
<thead>
<tr>
<th>String</th>
<th>Attribute Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>domainComponent</td>
</tr>
<tr>
<td>CN</td>
<td>commonName</td>
</tr>
<tr>
<td>OU</td>
<td>organizationalUnitName</td>
</tr>
<tr>
<td>O</td>
<td>organizationName</td>
</tr>
</tbody>
</table>

Table 1) Common RDN values in LDAP.

In ONTAP LDAP client configuration, it is possible to specify a DN for base, user, group, and netgroup LDAP searches.

In Figure 2, ADSI Edit is used to display the folder structure of the LDAP DNs.

Figure 2) LDAP DN folder structure.

Why Should You Specify a DN?

In Figure 2, numerous folders could contain users, groups, or netgroups. If the base DN of dc=domain,dc=netapp,dc=com represents the domain of domain.netapp.com, the LDAP client could certainly be pointed to use the base DN. However, all queries would potentially need to crawl each DN that is listed below dc=domain,dc=netapp,dc=com, depending on the search scope. That process could add latency to name service and authentication lookups, which could cause authentication failures, access issues, or client connectivity problems. To avoid issues, it is critical that name service queries be returned as quickly as possible. Specification of DNs in the user, group, and netgroup lookups can help achieve the necessary speed by filtering requests to the LDAP server.

For example, if a user exists in the DN of cn=Users,dc=domain,dc=netapp,dc=com, then the user DN client configuration field can be specified to search in that DN for user objects, rather than in the base DN of dc=domain,dc=netapp,dc=com. That setting eliminates the need to look in any of the other DNs that exist below the base. This approach is especially critical in large environments, where there can be hundreds or even thousands of DNs.
Multiple DNs

It is possible to specify multiple DNs for object searches. Therefore, users, groups, and netgroups can exist in multiple locations in LDAP, and you can still filter searches by DN. When you specify multiple DNs, be sure to enclose the entry in double quotes, or else the command fails.

```
cluster::*> ldap client modify -client-config DOMAIN -vserver SVM -user-dn cn=users,dc=domain,dc=netapp,dc=com;OU=cdot,dc=domain,dc=netapp,dc=com
```

Error: "OU=cdot,dc=domain,dc=netapp,dc=com" is not a recognized command

**Note:** Be sure that the entries use a semicolon separator between DNs. If you use a comma to separate the values, the cluster sees the entry as a single DN.

If a user exists with the same user name in multiple DNs but has different numeric UIDs, then NFS lookups should work well, provided that those numeric UIDs don’t exist elsewhere in the LDAP environment.

NFS Numeric ID Operations

NFSv3 uses only numeric IDs for access unless NTFS ACLs are involved. For instance, if a user accesses an NFSv3 mount as user 1234 and the underlying permissions are UNIX style, then all ONTAP cares about is the 1234 UID to allow or to deny access.

However, NFSv4.x can have two different approaches for UID-to-name resolution. By default, NFSv4.x uses the numeric IDs, similar to NFSv3 use of the `v4-numeric-ids` option.

```
[-v4-numeric-ids {enabled|disabled}] - NFSv4 Support for Numeric Owner IDs
This optional parameter specifies whether to enable the support for numeric string identifiers in NFSv4 owner attributes. The default setting is enabled at the time of creation.
```

Alternatively, NFSv4.x can use ID strings, in which a numeric ID must resolve to a user name, and that username must then exist in the same domain on the client and server. For instance, UID 1234 must map to a user name, such as `user@DOMAIN.COM`. Otherwise, that user gets squashed to a `nobody` user specified in the client’s NFSv4.x configuration files. To force ID string resolution, `v4-numeric-ids` must be disabled.

Example of Identical User Names, Different UIDs

For example, if a user named `v4user` exists locally on the client machine and in LDAP with two different numeric UIDs, then the behavior of file creation and access varies depending on the `ns-switch` configuration (which controls what UID is used) and NFS server settings.

If `ns-switch` uses LDAP first on the client, then the user becomes whatever LDAP has stored for the user information. In this example, the client’s `ns-switch` is set up for user resolution as follows:

```
passwd:     sss files
```

That setup means that users (passwd) use LDAP and then local files. It is similar to how multiple LDAP servers with the same user name and with different UIDs operate.

On the example client, following is the `v4user` when the LDAP client is disabled and local files are used:

```
# id v4user
uid=1005(v4user) gid=1005(v4user) groups=1005(v4user)
```

When the LDAP client is running, the user appears as follows:

```
# id v4user
uid=877(v4user) gid=10000(Domain Users) groups=10000(Domain Users)
```
If a file gets written when the LDAP client isn’t running, then the UID becomes 1005.

```shell
-rwx------ 1 1005 1005 23 Feb 3 16:38 v4user_file
```

If a file gets written when the LDAP client is running, then the UID becomes 877, even though the user name is identical across all the files.

```shell
-rwx------ 1 877 10000 22 Feb 3 16:39 v4user_file3
```

```shell
# ls -la | grep v4user
-rwx------ 1 v4user v4user 23 Feb 3 16:38 v4user_file
-rwx------ 1 v4user v4user 26 Feb 3 16:19 v4user_file2
-rwx------ 1 v4user Domain Users 22 Feb 3 16:39 v4user_file3
-rwx------ 1 v4user Domain Users 0 Feb 3 10:34 v4user_file4
```

When the v4-numeric-ids option is disabled on the NFS server, then only the user that can be resolved through LDAP is shown properly. That user is the only one that ONTAP knows how to translate for NFSv4.x access, even though the client knows about both users.

```shell
cluster::*> nfs server show -vserver DEMO -fields v4-numeric-ids
vserver v4-numeric-ids
------- --------------
DEMO    disabled
```

```shell
sh-4.2$ ls -la | grep v4user
-rwx------ 1 nobody nobody 23 Feb 3 16:38 v4user_file
-rwx------ 1 nobody nobody 26 Feb 3 16:19 v4user_file2
-rwx------ 1 v4user Domain Users 22 Feb 3 16:39 v4user_file3
-rwx------ 1 v4user Domain Users 0 Feb 3 10:34 v4user_file4
```

The issue with having multiple user names with the same name but different UIDs is that it affects access. Access depends on the permissions for the UID, not for the user name. Although the clients might see the same user name, because the underlying UIDs are different, access might be denied.

In the following example, UID 1005 (v4user) has ownership of v4user_file. Other users and groups have no access to that file.

```shell
-rwx------ 1 1005 1005 23 Feb 3 16:38 v4user_file
```

However, the user who accesses the NFS mount is actually UID 877.

```shell
sh-4.2$ id
uid=877(v4user) gid=10000(Domain Users) groups=10000(Domain Users)
```

As a result, access to the file that 1005 owns is denied, because v4user (877) is not the same as v4user (1005).

```shell
sh-4.2$ cat v4user_file
cat: v4user_file: Permission denied
```

Because this user is UID 877, then access to v4user_file3 succeeds, because the owner is also 877.

```shell
-rwx------ 1 877 10000 22 Feb 3 16:39 v4user_file3
```
To prevent confusion over access, disable `v4-numeric-ids` to allow the files to show as owned by `nobody`. However, ultimately, if you use multiple DNs, to achieve predictable access permissions, ensure that all UIDs and user names are unique.

Search Scopes

In addition to DNs, you can specify search scopes for LDAP queries. A scope is the starting point for LDAP queries and indicates at what depth from the base DN the search should occur. The following search scopes are valid for LDAP queries in ONTAP: base, subtree, and onelevel.

**base**

A base search scope indicates that LDAP searches should occur only for the specified base DN.

For example, if a user DN is set to `cn=users,dc=domain,dc=netapp,dc=com`, and a base search scope is specified, then LDAP search occurs only for `cn=users,dc=domain,dc=netapp,dc=com`. It does not include objects that are inside the DN. Because base is a very literal search scope, NetApp does not recommend it.

**subtree**

A subtree search scope searches all levels below the specified DN.

For example, if a user DN is set to `cn=users,dc=domain,dc=netapp,dc=com`, and a subtree search scope is specified, then LDAP search occurs for all objects below `cn=users,dc=domain,dc=netapp,dc=com`, including other containers. NetApp recommends subtree as a search scope in most cases, as long as the DN that you specify is at a low enough level to filter effectively.

In Figure 3, the DN of `dc=domain,dc=netapp,dc=com` has several containers beneath it. If a search scope of subtree is used, then LDAP queries look in each DN below `dc=domain,dc=netapp,dc=com`. You might prefer to specify the DN at a more granular level for subtree searches, such as `cn=users,dc=domain,dc=netapp,dc=com`.

Figure 3) LDAP DN containers.
onelevel
A onelevel search scope searches only at one level below the specified DN, including the DN itself, but does not search entries below that one level.

For example, if a user DN is set to `cn=users,dc=domain,dc=netapp,dc=com`, and a onelevel search scope is specified, then LDAP search occurs only in DNs that are one level below `cn=users,dc=domain,dc=netapp,dc=com`.

```bash
cluster::*> ldap client modify -client-config DOMAIN -vserver SVM -user-dn "cn=users,dc=domain,dc=netapp,dc=com" -user-scope onelevel
```

In ONTAP 8.3 and later versions, you can query the user ID from the advanced privilege with the following command:

```bash
cluster::*> getxsbbyname -vserver DEMO -username prof1 -node node1
```

In ONTAP 9.6 and later, you can also use the following command:

```bash
cluster::*> access-check authentication translate -vserver DEMO -unix-user-name prof1
```

If the DN is set at a level higher, the lookup fails.

```bash
cluster::*> ldap client modify -client-config DOMAIN -vserver SVM -user-dn "dc=domain,dc=netapp,dc=com" -user-scope onelevel
```

```bash
cluster::*> access-check authentication translate -vserver DEMO -unix-user-name prof1
```

Vserver: SVM (internal ID: 3)
Error: Acquire UNIX credentials procedure failed
   [  0 ms] Name 'prof1' not found in UNIX authorization source LOCAL
   [  0] Connecting to LDAP (NIS & Name Mapping) server 10.x.x.x
   [  5] Using a new connection to 10.x.x.x
   [  7] Name 'prof1' not found in UNIX authorization source LDAP
   [  7] Could not get a user ID for name 'prof1' using any NS-SWITCH authorization source
**[  7] FAILURE: Unable to retrieve UID for UNIX user prof1
```

Error: command failed: Failed to resolve user name to a UNIX ID. Reason: "SecD Error: object not found".

For filtering at a very granular level, onelevel search scopes are optimal. However, if the correct DNs are not specified, use of onelevel search scopes can result in lookup failures.

### 3.5 Centrify Integration and Considerations

Windows Active Directory LDAP servers that run Windows 2016 and later have deprecated the UNIX Attributes tab in the Properties dialog box. The preferred method is Windows PowerShell and/or advanced Attributes management from the Active Directory Users and Computers GUI.
Management of large environments with many users can be cumbersome, so third-party products, such as Centrify, attempt to simplify user authentication across Windows and Linux clients by providing a single-pane-of-glass approach to identity management.

ONTAP is fully compatible with any LDAP provider, as long as the provider follows the RFC 2307 standards.

To use ONTAP with a third-party LDAP provider such as Centrify, you must contact the LDAP administrator to get information about how the LDAP server presents users, groups, netgroups, and group memberships in their schemas. For instance, some LDAP providers use uid for user names; others might use uid to mean the numeric IDs for the users. In some cases, object classes might differ greatly between LDAP providers. Before you configure ONTAP as an LDAP client, you must find out the following schema attributes for users:

- User object classes
- User names
- Numeric UIDs
- Primary GIDs
- Home directory
- UNIX password
- Gecos

You must obtain the following information for groups and group memberships:

- Group object classes
- Numeric GID
- Group names
- Whether memberUid is populated for group memberships
- Whether the LDAP server is using RFC 2307bis for group memberships (which Active Directory can do)

For Windows Active Directory, you can use PowerShell to query LDAP users and groups for these attributes.

For UNIX LDAP servers, you can use utilities such as ldaps

After you have gathered the necessary information, use the steps in the “Create Custom LDAP Schemas” section of this document to create a custom schema to configure the Centrify LDAP lookups.

### 3.6 LDAP Referrals (Chase Referrals)

LDAP referrals provide a way for ONTAP to support environments in which UNIX users and groups exist in multiple LDAP servers. When LDAP referrals are disabled (default), ONTAP looks for a user in an LDAP server. If that user doesn't exist in the LDAP server, then the query is done and the lookup fails.

When LDAP referrals are enabled (referral-enabled true) and multiple LDAP servers are listed, if a user is not found on the first LDAP server, a referral is issued to other LDAP servers in the list to see whether the user exists there. LDAP queries take longer to perform these additional searches, and the amount of extra time needed depends on network latency, size of the LDAP schemas, and number of LDAP servers specified. If searches are timing out with LDAP referrals enabled, you can increase the LDAP query timeout from the default of 3 seconds up to a maximum of 10 seconds with the -query-timeout option.

LDAP referrals are not supported when LDAP over StartTLS is used.
4 Configuration

This section covers only the LDAP client configuration for NetApp ONTAP. For NFS client LDAP configuration, see the client documentation or review TR-4073. For LDAP server configuration, see the documentation for your specific LDAP server.

Because ONTAP System Manager configuration support is limited mostly to default schemas, is missing some configuration values, and focuses mostly on Microsoft Active Directory LDAP configuration, this TR covers CLI configuration only.

LDAP configuration generally consists of the following steps:

1. Gather configuration information (server IP addresses and names, LDAP schema information, bind information, and so on).
2. Select or create an LDAP schema.
3. Create the LDAP client configuration.
4. Create the LDAP configuration.
5. Modify ns-switch to use LDAP.
6. Test LDAP.

4.1 LDAP Environment Information—Preconfiguration

Before you configure ONTAP as an LDAP client, you should gather some information to make it easier for you to set up everything.

Before You Start

Before you set up LDAP on a production SVM, it might be better to create a new test SVM to get the LDAP configuration correct before you deploy it in a production SVM. If you try to deploy the LDAP client configuration in a production environment, you run the risk of breaking user authentication if LDAP isn’t working properly.

A new test SVM requires only a data LIF that can contact the same LDAP servers that will be used in the production environment.

Success criteria will be based on admin-generated LDAP calls, which are covered in the section “Test LDAP Functionality.”

LDAP Security Decisions

LDAP connectivity can use encryption for binds and LDAP searches. ONTAP offers a few options to secure LDAP connections, including:

- Binding as a CIFS/SMB server (by using the CIFS/SMB machine account to bind through NT LAN Manager [NTLM] or Kerberos)
- SMB signing and sealing (when you use Windows Active Directory for LDAP)
- StartTLS or LDAP over SSL (only one can be enabled at a time)

It is important to coordinate with the LDAP administrator to find out what security methods are being used. In some cases, security for LDAP is being enforced, so LDAP client configuration in ONTAP would need to use the provided security method simply to get LDAP working.

For information about configuring LDAP over SSL or StartTLS in ONTAP for LDAP, see section 4.4, “Configure Secure LDAP.”
StartTLS (Transport Layer Security) Versus LDAP Over SSL (LDAPS)

ONTAP offers both LDAPS (LDAP over SSL that uses port 636) and LDAP that uses StartTLS (port 389). LDAPS is considered legacy at this point, with RFC 1777 having been published in 1995. LDAP over StartTLS was introduced with RFC 2830 in 2000 and was combined into the LDAPv3 standard with RFC 4511 in 2006. After StartTLS was made a standard, LDAP vendors began to refer to LDAPS as deprecated.

Originally, ONTAP supported only StartTLS for LDAP encryption, but popular demand brought LDAPS into ONTAP 9.5 and later versions. In most cases, you want to use StartTLS (or LDAP signing and sealing if you use Windows Active Directory) as your LDAP encryption, even though StartTLS uses the plaintext, well-known LDAP port 389. With StartTLS, after the initial LDAP connection has been made, a StartTLS OID is exchanged, certificates are compared, and then all traffic is encrypted by using TLS. The packet capture shown in Figure 4 shows the LDAP bind, StartTLS handshake and subsequent TLS-encrypted LDAP traffic.

Figure 4) Packet trace of StartTLS LDAP traffic.

LDAPS also uses certificate exchange and encrypts by using TLS. A trace shows traffic only over port 636. Figure 5 shows the traffic in a packet capture from an LDAPS conversation.
There are two main differences between LDAPS and StartTLS:

- **StartTLS** is part of the LDAP standard; LDAPS is not. As a result, LDAP library support might vary, and functionality might or might not work in all cases.

- If encryption fails, StartTLS allows the configuration to fall back to regular LDAP. LDAPS does not. As a result, StartTLS offers some flexibility and resiliency, but it also presents security risks if it is misconfigured.

Choosing StartTLS or LDAPS is a matter of preference in most cases, but for full standard compliance, NetApp recommends StartTLS.

**Note:** With ONTAP LDAP clients, you can set only LDAPS (by setting port 636) or StartTLS (`-use-start-tls true` and port 389). You cannot enable them together.

**Security Considerations with StartTLS**

StartTLS enables administrators to fall back to regular LDAP traffic if they want to, but for security purposes, most LDAP administrators do not want to allow it. NetApp recommends the following to secure StartTLS to help secure LDAP communication:

- Ensure that StartTLS is enabled and that certificates are configured.

- For internal environments, you can use self-signed certificates, but for external LDAP, use a certificate authority. For more information about certificates, see the Microsoft TechNet article [Difference Between Self Signed SSL & Certificate Authority](#).

- Prevent LDAP queries and binds that do not use StartTLS. Follow the configuration steps for the LDAP server from the LDAP server vendor.

- Restrict LDAP clients from sending plaintext credentials by setting the minimum bind level (`-min-bind-level`) on the LDAP client in ONTAP to SASL.

**Microsoft LDAP Channel Binding Requirement**

Because of a vulnerability with Windows Active Directory domain controllers, a default setting is being changed for Windows servers that could affect LDAP interaction with ONTAP. For details, see Microsoft Security Advisory [ADV190023](#).

Essentially, Microsoft Windows will start recommending that administrators enable LDAP signing and channel binding. If the LDAP client supports channel binding tokens and LDAP signing, channel binding and signing will be required and registry options will be set by the new Microsoft patch.
Impact for ONTAP

The impact for ONTAP will be the same as for any LDAP client. For things that the client supports, those things will be required. If the security setting is not supported, no changes will be necessary. However, for ONTAP, it means two things:

- ONTAP does not currently support channel binding, so no changes will be necessary.
- ONTAP does support LDAP signing. Therefore, when the patch is applied to Windows, if LDAP signing is not enabled, ONTAP LDAP communication will fail for both SMB and LDAP UNIX identity management.

Remediation Steps

After the Windows patch has been applied, if you have explicitly set the LDAP signing requirement to off, then no remediation is necessary. The patch will not overwrite the explicit change. However, if you have never set that option, then the patch will change the option to on. So that ONTAP can handle those changes, if you are using CIFS/SMB, you should set the CIFS/SMB server option --session-security-for-ad-ldap to sign CIFS/SMB. If you use LDAP with Active Directory for UNIX user and group lookups, you also need to use either StartTLS (--use-start-tls), LDAPS (port 636 and certificates), or LDAP signing and sealing (--session-security sign).

Note: Bug 1289739 plans to address this issue without user interaction in a later release of ONTAP.

For more information, see the NetApp Knowledgebase article titled Impact of "Microsoft Security Advisory ADV190023" for Remote Authentication using LDAP.”

DNS

LDAP servers often make use of DNS, especially when Active Directory is in use.

Before you configure the SVM for LDAP, you should create the DNS configuration that contains records for the LDAP servers and services. DNS calls require that at least one data or SVM management LIF in the SVM can contact the DNS servers.

This configuration is performed at the SVM level.

```
cluster::> dns create ?
  [-vserver] <vserver name>          Vserver
  [-domains] <text>, ...             Domains
  [-name-servers] <IP Address>, ...  Name Servers
  [[-timeout] {1..5}]                Timeout (secs) (default: 2)
  [-attempts {1..4} ]                Maximum Attempts (default: 1)
  [ -skip-config-validation [true] ]  Skip Configuration Validation
```

For information about how to test whether DNS is working, see section 5.3, “Troubleshooting Tools.”

LDAP Server Information

The LDAP server information that you should gather includes:

- Type of LDAP server (Windows Active Directory, Red Hat, and so on)
- LDAP server IP address, host name, or Active Directory domain
- Search DN and scope information
- DNS SRV record information for LDAP
- Whether the LDAP servers are behind a load balancer/network address translation (NAT) address
- LDAP server port
- LDAP server security level for binds (anonymous, simple, or SASL)
- Whether LDAP over SSL is being used
• Whether chase referrals are in use (for more information about referrals, see RFC 4511)

LDAP Schema Configuration

Before you can configure SVMs in ONTAP as LDAP clients, you must choose and configure a schema. As mentioned previously, ONTAP has several default schema templates to assist you. You can use Table 2 as guidance for which schema template to use. In most cases, you can use the defaults. If variations exist, you can use these schema templates as a starting point by copying them to custom LDAP schemas and then modifying the desired specific attributes.

Note: These schema templates apply most of the time, but they might vary for third-party LDAP management tools such as Centrify or Quest.

Table 2) ONTAP LDAP schema templates and corresponding LDAP servers.

<table>
<thead>
<tr>
<th>LDAP Schema</th>
<th>LDAP Server Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD-SFU</td>
<td>Windows 2003 and earlier</td>
</tr>
<tr>
<td>AD-IDMU</td>
<td>Windows 2003 R2 and later</td>
</tr>
<tr>
<td>MS-AD-BIS</td>
<td>Windows 2003 R2 and later</td>
</tr>
<tr>
<td></td>
<td>(For more information, see the “RFC 2307bis” section.)</td>
</tr>
<tr>
<td>RFC 2307</td>
<td>Most UNIX/Linux-based LDAP servers (such as Red Hat and Apple)</td>
</tr>
</tbody>
</table>

Before you select a schema, verify whether the LDAP schema attributes that are being used match the schemas that ONTAP provides. To verify, you might have to contact the team who is managing the LDAP environment. You might also be able to query the schema through ldapsearch commands (by dumping the output of a user and group to see the attributes) or through PowerShell (if you are using Microsoft Active Directory). Consult your LDAP administrator for information.

Example of ldapsearch to dump a user's information:
Examples of Common LDAP Searches

Example of PowerShell commands to dump a user's or a group's information:

C:\> Get-ADUser -Identity [username] -Properties *
C:\> Get-ADGroup -Identity [groupname] -Properties *

Attributes Versus Object Classes

In the LDAP schema output, you see two different types of values that you can specify: attributes and object classes.

Object classes are how the LDAP queries look for specific objects. For example, if you are looking for a user object, then the object class that was defined tells the query to look for all objects in that class. Attributes are the values in the actual objects that make each object unique.

For example, multiple users are in the same object class, but each user has a unique name and numeric UID.

The output from the LDAP queries helps you select the proper LDAP schema to use. Object classes vary depending on the LDAP server in use. See Table 3.
### Table 3) LDAP schema object classes.

<table>
<thead>
<tr>
<th>Value</th>
<th>LDAP Schema Object Class (LDAP Schema Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User/posixAccount object class</td>
<td>User (Active Directory)*  posixAccount (RFC 2307)*</td>
</tr>
<tr>
<td>Group/posixGroup object class</td>
<td>Group (Active Directory)*  posixGroup (RFC 2307)*</td>
</tr>
<tr>
<td>NIS netgroup object class</td>
<td>nisNetgroup (all default schemas)*</td>
</tr>
<tr>
<td>groupOfUniqueNames object class</td>
<td>groupOfUniqueNames (RFC 2307)*  Group (Active Directory)*</td>
</tr>
<tr>
<td>windowsToUnix name mapping object class</td>
<td>posixAccount (RFC 2307)*  User (Active Directory)*</td>
</tr>
<tr>
<td>NIS object class</td>
<td>nisObject (all default schemas)*</td>
</tr>
</tbody>
</table>

* Preferred or default value.

The attribute values that you are most concerned with are listed in Table 4.

### Table 4) LDAP schema attributes.

<table>
<thead>
<tr>
<th>Value</th>
<th>LDAP Schema Attribute (LDAP Schema Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>UID (all default schemas)*  Name (Active Directory)  GivenName (Active Directory)  sAMAccountName (Active Directory)</td>
</tr>
<tr>
<td>Group name</td>
<td>CN (all default schemas)*  Name (Active Directory)  sAMAccountName (Active Directory)</td>
</tr>
<tr>
<td>Numeric UID</td>
<td>uidNumber (all default schemas)*</td>
</tr>
<tr>
<td>Numeric GID</td>
<td>uidNumber (all default schemas)*</td>
</tr>
<tr>
<td>Home directory</td>
<td>homeDirectory (RFC 2307)*  unixHomeDirectory (Active Directory)*</td>
</tr>
<tr>
<td>Group membership</td>
<td>memberUid (RFC 2307)*  memberUid (Active Directory 2008 and earlier)  Member (Active Directory 2008 R2 and later)*  UniqueMember (RFC 2307bis)*</td>
</tr>
<tr>
<td>Windows-to-UNIX name mapping</td>
<td>sAMAccountName (Active Directory)*  windowsAccount (RFC 2307)*</td>
</tr>
<tr>
<td>(for asymmetric name mapping)</td>
<td></td>
</tr>
<tr>
<td>NIS Map Name</td>
<td>nisMapName (all default schemas)*</td>
</tr>
<tr>
<td>NIS Map Entry</td>
<td>nisMapEntry (all default schemas)*</td>
</tr>
<tr>
<td>NIS netgroup triple</td>
<td>nisNetgroupTriple (all default schemas)*</td>
</tr>
</tbody>
</table>
### Value | LDAP Schema Attribute (LDAP Schema Type)
---|---
NIS netgroup member | memberNisNetgroup (all default schemas)*

*Preferred or default value.

#### 4.2 LDAP Environment Configuration

Now that you have gathered the necessary information, you can start to create your LDAP configuration. Because some steps depend on other steps, you must follow a specific order. Generally, you follow this order:

1. Select, create, or configure an LDAP client schema.
2. Create the LDAP client configuration for the SVM.
3. Enable LDAP for use with the SVM.
4. Modify `ns-switch` to use LDAP.
5. Test LDAP lookups.

### LDAP Schemas

First, you need an LDAP schema, because LDAP client configuration requires a schema during the creation step. To ensure that you have collected the proper information before you select a schema, review the section "LDAP Schema Configuration" in this report.

**Select an LDAP Schema**

After you have figured out what the appropriate LDAP schema might be, you can compare and contrast LDAP schema templates in ONTAP with what you have in your LDAP server environment.

**Note:** You can find LDAP schema template examples in the appendix of this report, under “LDAP Schema Templates.”

If the available schema templates have what you need, then you can move on to the LDAP client configuration step. If you need a more customized LDAP schema, select the LDAP schema template that is closest to what you need and continue to the next section, “Create Custom LDAP Schemas.”

**Create Custom LDAP Schemas**

The default templates are read-only, so if you need to use LDAP schemas that do not exist in the available default templates, you must create a new schema. To create a new schema, you can copy a schema from a template to a new LDAP client schema.

To copy an LDAP schema template to a new LDAP schema for modification and customization, use the following command:

```
cluster::*> ldap client schema copy ?
[ -vserver <vserver name> ] *Vserver
[-schema] <text (size 1..32)> *Schema Template
[-new-schema-name] <text (size 1..32)> *New Schema Template Name
```

After you have a new schema to work with, you can use the LDAP client schema modify commands to make changes.

```
cluster::*> ldap client schema modify ?
[ -vserver <vserver name> ] Vserver (default: cluster)
[-schema] <text (size 1..32)> Schema Template
[[ -comment] <text> ] Comment
[ -posix-account-object-class <text> ] RFC 2307 posixAccount Object Class
[ -posix-group-object-class <text> ] RFC 2307 posixGroup Object Class
[ -nis-netgroup-object-class <text> ] RFC 2307 nisNetgroup Object Class
```
In most cases, you do not need a custom schema. You can use the default schemas to begin with.

**LDAP Client Configuration**

Now that you have a valid LDAP schema to work with, you can create the LDAP client configuration. This configuration defines the necessary parameters for connecting and querying your LDAP servers. To construct the client configuration for the ONTAP SVM, use the data that you gathered according to the "LDAP Server Information" section of this report.

Table 5 lists the LDAP client configuration options for modification from the diag privilege.

### Table 5) LDAP client configuration options (diag privilege).

<table>
<thead>
<tr>
<th>Configuration Option</th>
<th>What It Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>-vserver</td>
<td>Specify the SVM (vserver) that owns the LDAP configuration.</td>
</tr>
<tr>
<td>-client-config</td>
<td>This option is the name of the client configuration.</td>
</tr>
<tr>
<td>-ldap-servers</td>
<td>This option is the list of LDAP servers or host names. If you use Microsoft Active Directory LDAP, use -ad-domain instead.</td>
</tr>
<tr>
<td>-servers (deprecated)</td>
<td>Use -ldap-servers instead.</td>
</tr>
<tr>
<td>-ad-domain</td>
<td>This option defines the Active Directory domain to be used for LDAP server lookups and name resolution. This option causes ONTAP to use DNS SRV record lookups for Active Directory LDAP servers. If you use Linux/UNIX LDAP servers, use -ldap-servers. If you want to specify LDAP servers to use in Active Directory, use –preferred-ad-servers in addition to –ad-domain.</td>
</tr>
<tr>
<td>-bind-as-cifs-server</td>
<td>Use this option only if a CIFS/SMB server is present in the SVM. Binding as a CIFS server means that the LDAP searches use the credentials of the CIFS/SMB machine account to log in to Active Directory for LDAP queries.</td>
</tr>
<tr>
<td>-schema</td>
<td>This option defines the LDAP schema that you want to use.</td>
</tr>
<tr>
<td>Configuration Option</td>
<td>What It Means</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-port</td>
<td>This option allows the LDAP port to be changed. The default for LDAP is 389; for LDAPS, the port is 636. If you want global catalog searches in Active Directory LDAP, use port 3268.</td>
</tr>
<tr>
<td>-query-timeout</td>
<td>This option defines how long a query runs before it times out. The default is 3 seconds.</td>
</tr>
<tr>
<td>-min-bind-level</td>
<td>This option defines the minimum bind security that is allowed for LDAP binds.</td>
</tr>
</tbody>
</table>
| -bind-dn                 | This option defines the user that is used for LDAP binds/logins. The format can be:  
  - Username  
  - Username@domain.com (Active Directory)  
  - DOMAIN\username (Active Directory)  
  - DN=username,DN=domain,DN=com |
<p>| -base-dn                 | This option defines the base search DN for LDAP queries. If you use -ad-domain, it is automatically set to the Active Directory domain DN. For example, domain.com becomes the base DN DC=domain,DC=com. |
| -base-scope              | This option defines the base search scope. It defaults to subtree.                                                                        |
| -user-dn                 | This option defines the search DN for the User object class. Use this option if filtering is needed to speed up queries. If you leave this option blank, ONTAP uses the base search DN. |
| -user-scope              | This option defines the search scope for the User object class. The default is subtree.                                                   |
| -group-dn                | This option defines the search DN for the Group object class. Use this option if filtering is needed to speed up queries. If you leave this option blank, ONTAP uses the base search DN. |
| -group-scope             | This option defines the search scope for the Group object class. The default is subtree.                                                   |
| -netgroup-dn             | This option defines the search DN for the Netgroup object class. Use this option if filtering is needed to speed up queries. If you leave this option blank, ONTAP uses the base search DN. |
| -netgroup-scope          | This option defines the search scope for the Netgroup object class. The default is subtree.                                                   |
| -use-start-tls           | This option defines whether start-tls is used to secure LDAP. StartTLS uses port 389 and is not LDAP over SSL (LDAPS). For LDAPS, change the LDAP port to 636. |
| -is-netgroup-byhost-enabled | This option defines whether netgroups are queried by netgroup name (set to false) or by host name (set to true). For further guidance, see the “Use of LDAP to Host Netgroups” section in this document. |
| -netgroup-byhost-dn      | This option defines the search DN for the Netgroup-by-host object class. Use this option if filtering is needed to speed up queries. If you leave this option blank, ONTAP uses the base search DN. |</p>
<table>
<thead>
<tr>
<th>Configuration Option</th>
<th>What It Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>-netgroup-byhost-scope</td>
<td>This option defines the search scope for the Netgroup-by-host object class. The default is subtree.</td>
</tr>
<tr>
<td>-session-security</td>
<td>This option defines the level of session security. Sign, seal, sign and seal, and none are valid options.</td>
</tr>
<tr>
<td>-skip-config-validation</td>
<td>Config validation attempts to connect and test LDAP servers before configurations are applied. This option allows you to skip those steps. The default is set to true.</td>
</tr>
<tr>
<td>-referral-enabled</td>
<td>This option defines whether ONTAP uses the chase referrals functionality with LDAP servers, which allow LDAP queries to connect to other LDAP servers if a requested object does not exist in the first LDAP server specified.</td>
</tr>
<tr>
<td>-group-membership-filter</td>
<td>This parameter specifies the custom LDAP search filter to be used when looking up group membership from an LDAP server. Examples of valid filters are (cn=<em>99), (cn=1</em>), (</td>
</tr>
</tbody>
</table>

You can find an example in the appendix section “Sample LDAP Client Configuration” later in this document.

**LDAP Client Configuration—SVM Scopes**

When you create an LDAP client configuration in LDAP, you have one of two choices:

- **Explicitly set the -vserver option.** This option makes the client configuration available only to the specified SVM, which can provide secure multitenancy functionality in environments that are shared by many different customer bases.

- **Leave the -vserver option blank.** This option creates the client configuration and makes it available to all SVMs in the cluster. This option is useful when you need to share LDAP client configurations across multiple SVMs in your environment.

**Enable LDAP**

Now that you have an LDAP client configuration, you must run the `ldap create` command on the SVM to enable LDAP. To allow a check to occur on the configuration, NetApp recommends that you set `-skip-config-validation` to false.

```
cluster::> ldap create ?
    [-vserver] <vserver name>          Vserver
    [-client-config] <text>            LDAP Client Configuration
    [[-skip-config-validation] [true|false]]  Skip Configuration Validation
```

This setting simply allows the SVM to use LDAP; it does not participate in authentication requests until the SVM’s ns-switch settings are modified.

**Modify the SVM Name Service Switch (ns-switch)**

ONTAP does not start using LDAP for name lookups until LDAP is specified in the SVM ns-switch configuration, and it uses LDAP only for the databases that are specified by the ns-switch commands. You can specify LDAP as a valid name service for:

- Passwd
- Group
• Namemap
• Netgroup

The following covers which name service databases are available for use. If you do not plan to use LDAP for certain services, do not enable it. For example, name mapping can incur delays if LDAP is specified as a valid name service source for name mapping rule, but is not configured to do name mappings. For more information, review the section “Use of LDAP for Name Mapping.”

cluster::> ns-switch modify?
(vserver services name-service ns-switch modify)
   -vserver <vserver name> Vserver
   [-database] {hosts|group|passwd|netgroup|namemap} Name Service Switch Database
   [-sources] {files|dns|ldap|nis}, ... Name Service Source Order

Test LDAP Functionality

Now that the configuration is complete and the ONTAP SVM is set up to use LDAP for name service and identity requests, you must test functionality. To do this testing, operate out of diag privilege. To enter diag privilege, use:

cluster::> set diag

This setting makes all diagnostic and troubleshooting commands available.

LDAP Connectivity

These commands help you troubleshoot whether LDAP servers can be reached and whether you can connect to them. However, in ONTAP 9.5 and later, if there is a connectivity or configuration issue, LDAP and DNS configuration commands fail because ONTAP automatically runs checks before it applies configuration changes. For example, if you change something on the LDAP client, ONTAP checks network connectivity, LDAP bind, and searching for the configured DNs in the client configuration. If any of those checks fail, the command to change the configuration fails. You can bypass this check with the -skip-config-validation true option.

DNS Lookups

To look up LDAP servers by host name or IP addresses by using DNS calls, use the following commands:

getxxbyyy gethostbyname -node [node1] -vserver [SVM] -hostname [ldap.ntap.local]
getxxbyyy gethostbyaddr -node [node1] -vserver [SVM] -ipaddress [10.10.10.10]

You can also use the following for forward lookups:

diag secd dns forward-lookup -node [node1] -vserver [SVM] -hostname [hostname]

To look up SRV records, use:

diag secd dns srv-lookup -node [node1] -vserver [SVM] -lookup-string [_ldap._tcp.ntap.local]

Network Pings

Pinging the LDAP servers can be a way to see whether the SVM LIFs can reach LDAP. However, keep in mind that some networks block Internet Control Message Protocol (ICMP) traffic, so pings might not work properly. When pinging, be sure to define the SVM name and data LIF name so that the ping uses the LIFs that are participating in the LDAP traffic.

cluster::*> ping?
   { -node <nodename> Node
     | -lif <lif-name> } Logical Interface
     -vserver <vserver> Vserver
LDAP Connection Test

To test LDAP (and other name service server) connections, use diag secd connections test.

```
cluster::*> diag secd connections test -node [node1]-vserver [SVM]
```

To view the connections, use:

```
cluster::*> diag secd connections show ?
    [-node] <nodename> *Node
    [-vserver] <vserver> *Vserver
    [-type] <text> *Cache type (lsa,netlogon,ldap-ad,ldap-nis-namemap)
    [-key <text>] *Connection key
```

To clear the connections cache, use:

```
cluster::*> diag secd connections clear ?
    [-node] <nodename> *Node
    [-vserver] <vserver> *Vserver
    [-type] <text> *Cache type (lsa,netlogon,ldap-ad,ldap-nis-namemap)
    [-key <text>] *Connection key
```

Alternatively, you can use the ldap check command.

```
cluster::*> ldap check -vserver DEMO
```

Client Configuration Name: DEMO
Vserver: DEMO
LDAP Status: up
LDAP Status Details: Successfully connected to LDAP server "10.x.x.x".
LDAP DN Status Details: All the configured DNs are available.

LDAP Name Lookup and Group Membership

Because of changes in how name services operate, LDAP name lookups occur at the SVM level in ONTAP 9.3 and later. Therefore, you should perform LDAP tests by using getxxbyyyy commands; see the section “GetXXbyYY.”

Following is an example of group membership for a user:

```
cluster::*> getxxbyyyy getgrlist -node nodel -vserver DEMO -username prof1 -show=granular-err true -use-cache false -show-source true (vserver services name-service getxxbyyyy getgrlist)
Source used for lookup: LDAP
pw_name: prof1
Groups: 1101 1201 1202 1203 1220
NIS:
Error code: NS_ERROR_NONE
Error message: No error
LDAP:
Error code: NS_FOUND
Error message: Entry found
DNS:
Error code: NS_ERROR_NONE
Error message: No error
FILES:
```
You can also carry out name lookups and translations through `secd`.

```
cluster::* > diag secd authentication show-ontap-admin-unix-creds ?
[ -node <nodename> ] *Node (default: ontap9-tme-8040-01)
[ -vserver ] <vserver> *Vserver
[ -unix-user-name ] <text> *Unix User Name
[ -uid ] <integer> } *Unix User ID

cluster::* > diag secd authentication translate ?
[ -node <nodename> ] *Node Name (default: ontap9-tme-8040-01)
[ -vserver ] <vserver> *Vserver Name
[ -uid ] <integer> } *UNIX User ID
[ -gid ] <integer> } *UNIX Group ID
[ -uid ] <text> } *Windows SID
[ -unix-user-name ] <text> *UNIX User Name
[ -unix-group-name ] <text> *UNIX Group Name
[ -win-name ] <text> } *Windows Name
```

In ONTAP 9.6 and later, you can use the `advanced privilege` `vserver services access-check` commands found in the ONTAP CLI Commands for LDAP Troubleshooting” section..

```
cluster::* > vserver services access-check ?
    authentication> *Check Authentication Information
    dns> *Check DNS Lookups
    name-mapping> *Check Name Mapping Operations
    server-discovery> *Check Server Discovery Information

cluster::* > vserver services access-check authentication show-ontap-admin-unix-creds -vserver
    DEMO -unix-user-name prof1
    User Id: 1100
    Group Id: 1101
    Home Directory: /home/prof1
    Login Shell: /bin/sh
```

**Name Mapping in Multiprotocol NAS Environments**

When you use NFS and CIFS/SMB on the same sets of data, accessing those files and folders require name mapping between UNIX and Windows users (and vice versa). The name mapping methodology depends on the security styles of the volumes and qtrees that are used in the environment. For example, if a volume uses NTFS security and a user accesses the volume through NFS, then the user must map to a Windows user to ascertain the appropriate permission levels because NFS does not understand NTFS ACLs.

To see what UNIX user maps to a specific Windows user, you can use the following command

```
cluster::* > diag secd name-mapping show -node [node] -vserver [SVM] -direction win-unix -name
    [username or DOMAIN\username]
```

In ONTAP 9.6 and later, you can use “`access-check`.”

```
cluster::* > access-check name-mapping show -vserver [SVM] -direction win-unix -name [username]
```

For UNIX-to-Windows name mapping, use:

```
cluster::* > diag secd name-mapping show -node [node] -vserver [SVM] -direction unix-win -name
    [username]
cluster::* > access-check name-mapping show -vserver [SVM] -direction unix-win -name [username]
```

For Kerberos service principal name (SPN)-to-UNIX users (for Kerberized NFS), use:
Name mapping values gather information from one the following places:

- **1:1 default name mappings in ONTAP** (user always maps to user, provided that both names can be found in name services)
- Name mapping rules (either locally or in LDAP)
- Default Windows or UNIX users (set at the CIFS and NFS server options -default-unix-user and -default-win-user, respectively)

Name mapping is only part of the equation, however. When you use a multiprotocol environment, you must ensure that all the appropriate group memberships are being populated in these requests. To query all the information about a user, use the following command:

```
cluster::*> diag secd name-mapping show -node [node] -vserver [SVM] -direction krb-unix -name [SPN]

cluster::*> access-check name-mapping show -vserver [SVM] -direction krb-unix -name [SPN]
```

4.3 **LDAP That Uses Active Directory for UNIX Identity Management**

The use of Microsoft Active Directory for UNIX identity management offers several benefits, including:

- Native Kerberos (Key Distribution Center [KDC]) integration
- Native LDAP replication and redundancy
- Forest-level replication through global catalog searches
- Support for trusted domains with LDAP operations

The following sections cover some of these concepts in deeper detail.
Domain Controller Redundancy and Replication

Active Directory, by default, replicates its databases to domain controllers every 15 minutes. All objects in the domain are copied across multiple locations, including user and computer objects and their attributes. Therefore, by having more than one domain controller in a domain, it is possible to eliminate single points of failure in LDAP and Kerberos.

If an LDAP server fails, ONTAP moves on to additional LDAP servers when the primary LDAP server fails connectivity tests. The next server in line depends on how the LDAP client is configured:

- If a server list is specified, the next server in the list is used.
- If a load-balanced host name is used, the next server in the load balancer is used.
- If the Active Directory domain is used, then the next server that is provided in DNS service record (SRV) lookups is used.

Use of the Domain Controller as an Identity Management Server for UNIX

Microsoft Active Directory does not act as an LDAP UNIX identity management server natively. In newer Windows versions (Windows 2012 and later), the UNIX attributes exist in the schema but are not populated. Therefore, you no longer have to “extend the schema” as you had to do in Windows 2008 and earlier versions.

UNIX Attributes

To leverage UNIX users, groups, and netgroups, you must populate UNIX attributes with UNIX information. Windows 2012 and earlier versions offered a UNIX Attributes tab that could populate these entries through GUI interaction. See Figure 6.

Figure 6) UNIX Attributes tab in Windows 2012.
Windows 2016 and later versions have deprecated support for the UNIX Attributes tab. To manage UNIX attributes in Windows 2016 and later, you can use PowerShell or the Attribute Editor tab (see Figure 7 and the section "Modification of UNIX Attributes in Active Directory"). The Attribute Editor tab is also available in Windows 2012.

Figure 7) Attribute Editor tab in Windows 2016.

UNIX Attributes That Are Used by ONTAP for LDAP Queries

When ONTAP looks up a user, it uses standard LDAP search functionality.

**Note:** For more information, see this ldapsearch page.

The queries are constructed based on the user or group that comes into the NAS share. ONTAP collects this information and then creates the ldapsearch query.

For example, if an NFS user ID of 1234 tries to access an export with NFSv4.x or NTFS ACLs, ONTAP must translate that user ID to a UNIX name that can be resolved in the ACL. LDAP is used if it is specified in the ns-switch database.

ONTAP uses the attributes that are assigned in the configured LDAP schema (see "LDAP Schemas") on the SVM's LDAP client for the search. In the MS-AD-BIS schema, uid-number-attribute uses the value uidNumber, and the posix-account-object-class value is User.

```
cluster::*> ldap client schema show -schema MS-AD-BIS -fields posix-account-object-class,uid-number-attribute
vserver   schema   posix-account-object-class uid-number-attribute
----------  --------  -------------------------------------------  ----------------------
DEMO        MS-AD-BIS User                                  uidNumber
```

So, for numeric ID 1234, the LDAP search syntax is:

```
(& (objectClass=User)(uidNumber=1234))
```
This query can be simulated from ONTAP by using the GetXXbyYY commands that are listed in the “Troubleshooting Tools” section of this document.

Table 6, Table 7, and Table 8 list the standard or most common attributes that are used with Windows Active Directory UNIX identity management for users and groups, as based on the MS-AD-BIS schema. The MS-AD-BIS schema is the preferred LDAP schema for most standard Windows 2012 and later LDAP deployments. For information about custom schemas, see the “LDAP Schemas” section.

**UNIX User Attributes Used by ONTAP for LDAP Queries**

Table 6) Standard UNIX user attributes (MS-AD-BIS schema).

<table>
<thead>
<tr>
<th>LDAP Client Schema Attribute (ONTAP LDAP Client)</th>
<th>LDAP Attribute Value (LDAP Server)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-posix-account-object-class</td>
<td>User</td>
</tr>
<tr>
<td>-uid-attribute</td>
<td>uid</td>
</tr>
<tr>
<td>-uid-number-attribute</td>
<td>uidNumber</td>
</tr>
<tr>
<td>-gid-number-attribute</td>
<td>gidNumber</td>
</tr>
<tr>
<td>-gecos-attribute name</td>
<td>_unixHomeDirectory</td>
</tr>
<tr>
<td>-home-directory-attribute</td>
<td>name</td>
</tr>
<tr>
<td>-user-password-attribute</td>
<td>unixUserPassword</td>
</tr>
<tr>
<td>-login-shell-attribute</td>
<td>LoginShell</td>
</tr>
<tr>
<td>-windows-to-unix-attribute</td>
<td>sAMAccountName</td>
</tr>
</tbody>
</table>

**Group Attributes**

Table 7) Standard UNIX group attributes (MS-AD-BIS schema).

<table>
<thead>
<tr>
<th>LDAP Client Schema Attribute (ONTAP LDAP Client)</th>
<th>LDAP Attribute Value (LDAP Server)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-posix-group-object-class</td>
<td>Group</td>
</tr>
<tr>
<td>-cn-group-attribute</td>
<td>cn</td>
</tr>
<tr>
<td>-gid-number-attribute</td>
<td>gidNumber</td>
</tr>
<tr>
<td>-member-uid-attribute</td>
<td>memberUid</td>
</tr>
<tr>
<td>-group-of-unique-names-object-class</td>
<td>Group</td>
</tr>
<tr>
<td>-unique-member-attribute</td>
<td>Member</td>
</tr>
</tbody>
</table>

**Note:** For information about memberUid versus Member attributes/secondary groups, see the section “Secondary, Supplemental, and Auxiliary GIDs.”
NetApp Attributes

Table 8) Standard UNIX netgroup attributes (MS-AD-BIS schema).

<table>
<thead>
<tr>
<th>LDAP Client Schema Attribute (ONTAP LDAP Client)</th>
<th>LDAP Attribute Value (LDAP Server)</th>
</tr>
</thead>
<tbody>
<tr>
<td>nis-netgroup-object-class</td>
<td>nisNetgroup</td>
</tr>
<tr>
<td>cn-netgroup-attribute</td>
<td>Name</td>
</tr>
<tr>
<td>member-nis-netgroup-attribute</td>
<td>memberNisNetgroup</td>
</tr>
<tr>
<td>nis-netgroup-triple-attribute</td>
<td>nisNetgroupTriple</td>
</tr>
<tr>
<td>nis-object-class</td>
<td>nisObject*</td>
</tr>
<tr>
<td>nis-mapname-attribute</td>
<td>nisMapName*</td>
</tr>
<tr>
<td>nis-mapentry-attribute</td>
<td>nisMapEntry*</td>
</tr>
</tbody>
</table>

Note: Entries with * are used for netgroup.byhost.

Modification of UNIX Attributes in Active Directory

When you create a Windows user name or group, you can also associate UNIX attributes to that object in Active Directory. In modern Windows operating systems, you have two options to create and modify UNIX objects: with the GUI or with PowerShell.

Create and Modify UNIX Objects with the GUI

The most common way that administrators have managed Active Directory is through the Active Directory Users and Computers (ADUC) Microsoft Management Console (MMC). With the MMC, you can create users and groups and then modify them afterward to populate UNIX attributes for use with LDAP clients. When you create a user or group through ADUC, there is no way to populate UNIX attributes. Therefore, you must double-click the object afterward and use the Attribute Editor tab in Windows 2012 and later versions. The Attributes Editor tab is not visible by default; you must enable the Advanced Features view in the MMC. See Figure 8.

Figure 8) Enable Advanced Features in MMC to see the Attributes Editor tab.
When you have enabled the advanced features and have opened the user properties, you can then modify the UNIX attributes in Attributes Editor. For details on what attributes need to be edited, see the list of user and group attributes in Table 6 and Table 7, respectively.

For information about creating and editing netgroups in Active Directory LDAP, see the section “ONTAP Interaction with Active Directory LDAP for Netgroups.”

Create and Modify UNIX Objects with PowerShell

PowerShell also has native user and group cmdlets (New-ADUser, New-ADGroup) that enable you to create and manage users and groups through the CLI or automation. With PowerShell, you can use the –OtherAttributes option to set the UNIX attributes when the object is created.

For example, if you want to create a user named user1 with a uidNumber of 5555 and a gidNumber of 1101, you use the following PowerShell command:

```powershell
PS C:\> New-ADUser -SamAccountName user1 -UserName user1@NTAP.LOCAL -Name user1 -OtherAttributes @{'uid'="user1";'uidNumber'="5555";'gidNumber'="1101"} -Enabled 1 -PasswordNeverExpires 1 -AccountPassword (Read-Host -AsSecureString "password" -Force) -Force
```

From ONTAP, you can now query this user.

```
cluster::*> getxxbyyy getpwbyname -node ontap9-tme-8040-02 -vserver DEMO -username user1 -show-source true -use-cache false
    (vserver services name-service getxxbyyy getpwbyname)
Source used for lookup: LDAP
pw_name: user1
pw_passwd: 
pw_uid: 5555
pw_gid: 1101
pw_gecos: 
pw_dir: 
pw_shell: 
```

You can modify existing users and groups through Set-ADUser and Set-ADGroup, respectively. With these commands, you can use the –Add or –Replace options to add or to replace UNIX entries. In the following example, you can add a second UID or user name to user1:

```powershell
PS C:\> Set-ADUser -Identity user1 -Add @{'uid'="user1alt"}
```

Now your UNIX user can use user1 or user1alt to query for UID 5555.

```
cluster::*> getxxbyyy getpwbyname -node node2 -vserver DEMO -username user1 -show-source true
    (vserver services name-service getxxbyyy getpwbyname)
Source used for lookup: LDAP
pw_name: user1alt
pw_passwd: 
pw_uid: 5555
pw_gid: 1101
pw_gecos: 
pw_dir: 
pw_shell: 
```

```
cluster::*> getxxbyyy getpwbyname -node node2 -vserver DEMO -username user1alt -show-source true
    (vserver services name-service getxxbyyy getpwbyname)
Source used for lookup: LDAP
pw_name: user1alt
pw_passwd: 
pw_uid: 5555
pw_gid: 1101
pw_gecos: 
pw_dir: 
pw_shell: 
```
Secondary and Auxiliary Groups

In many cases, UNIX users are members of multiple groups outside of their primary group. LDAP can query for these groups, but Active Directory LDAP has two different ways to search for group membership: through memberUid or through Member.

RFC 2307—Standard Method (memberUid)

One way that Active Directory LDAP can show secondary groups for a user is by leveraging the memberUid attribute that exists on group objects in Active Directory. When a user is added to the memberUid list, then LDAP can return those groups during LDAP queries. The downside is that in Active Directory, that attribute does not autopopulate when a user is added to a Windows group. Manual intervention is required.

For example, if a Windows administrator adds user1 to Windows groups named group1, group2, and group3, then only the Member attribute is populated in Active Directory. Note in Figure 9 that group2 has no entries in the memberUid field, despite having entries in Member.

Figure 9) Addition of users to Windows groups.

Unless the memberUid for each of those groups is modified to add user1, a check for UNIX group membership for that user does not show those groups properly. The memberUid attribute is a multivalued string, meaning that more than one user can be added to the list.

In the following example in Figure 10, users are added to group1.
In ONTAP, you check which UNIX group memberships user1 has. Group1 (gidNumber 1201) has memberUid populated and is seen in the list, while group2 (gidNumber 1202) has no memberUid populated and does not appear in the list. RFC 2307bis functionality is disabled.

```bash
cluster::*> ldap client schema modify -schema DEMO -enable/rfc2307bis false
cluster::*> getxxbyyyy getgrlist -node ontap9-tme-8040-01 -vserver DEMO -username user1 -use-cache false
   (vserver services name-service getxxbyyyy getgrlist)
pw_name: user1
Groups: 1101 1201
```

**RFC 2307bis (Member)**

RFC 2307bis is another way to query for group memberships with users, and it happens to be the perfect fit for Windows Active Directory environments. Recall that in the preceding section, when you add users to Windows groups, Active Directory populates the Member attribute (see Figure 9) but does not populate the memberUid attribute (Figure 10). As a result, secondary group membership for UNIX users does not populate properly.

When RFC 2307bis is enabled, ONTAP queries for the -unique-member-attribute that is configured in the LDAP client schema. By default, the MS-AD-BIS schema uses Member for that attribute.

On the same SVM with the same configuration that is exemplified in the previous section, you enable RFC 2307bis support and make no other changes to the user or group membership. Then you run the same group list command and see the proper group membership.

```bash
cluster::*> ldap client schema modify -schema DEMO -enable/rfc2307bis true
cluster::*> getxxbyyyy getgrlist -node ontap9-tme-8040-01 -vserver DEMO -username user1 -use-cache false
   (vserver services name-service getxxbyyyy getgrlist)
pw_name: user1
Groups: 1101 1201 1202 1203
```

For more information, see the section “RFC 2307bis.”
Use of Trusted Domains in a Forest for UNIX Identity Management

With Active Directory, you can set up a trust between two domains in a forest and have child domains below those parent domains. The default behavior of these environments is to use LDAP referrals to query additional LDAP servers when the first LDAP server does not have the user or group. ONTAP 9.5 and later versions support referrals. For more information about referrals, see section 3.6, “LDAP Referrals (Chase Referrals).”

For troubleshooting commands for domain trusts, see section 5.3, “Troubleshooting Tools.”

Active Directory Global Catalog Searches

If you are using Windows Active Directory for UNIX identity management, you can leverage the global catalog to populate UNIX attributes, to replicate across the domain forest, and to query in ONTAP over port 3268. In this setup, you can have multiple trusted domains in the same forest, all with unique UNIX users and groups, that replicate up to the top level of the forest. This approach enables ONTAP to search at a forest level and bypasses the need for LDAP referrals.

Replicate New Attributes to the Global Catalog

UNIX attributes are not replicated to the global catalog by default; therefore, they cannot be searched in the global catalog until you tell Active Directory to replicate them. Figure 11 shows an example of a trusted domain setup that you can use for LDAP lookups with global catalog searches.
How to Configure Attributes to Replicate to the Global Catalog

Active Directory uses a back-end schema to control how objects operate. This schema can be modified, but modification requires special steps. It is best to contact Microsoft when you modify the schema, but you can use the following steps to modify the schema attributes. This example was performed on a Windows 2012 R2 Active Directory Domain Controller, so be sure to verify that these steps work with your version of Windows.

You can modify schema attributes by using ADSI Edit and by connecting to the Schema Naming Context. See Figure 12.
After you have connected to the Schema Naming Context, you can navigate down through the `CN=Schema, CN=Configuration, DC=NTAP, DC=local` folder to the attributes. Attributes appear as `CN=attributename`, such as `CN=uid`. See Figure 12.

**Figure 12** Connecting to the Schema Naming Context.

After you find the attribute that you want to replicate to the global catalog, double-click it or right-click and select Properties. Then navigate to the `isMemberOfPartialAttributeSet` value. Double-click that value and toggle the option to True and then click OK and then Apply. See Figure 13.

**Figure 13** Schema Naming Context format.

After you find the attribute that you want to replicate to the global catalog, double-click it or right-click and select Properties. Then navigate to the `isMemberOfPartialAttributeSet` value. Double-click that value and toggle the option to True and then click OK and then Apply. See Figure 14.
You can also modify the attributes by using PowerShell and the following command:

```powershell
PS C:\> Set-ADObject 'CN=uidNumber,CN=Schema,CN=Configuration,DC=NTAP,DC=local' -Replace @{'isMemberOfPartialAttributeSet="TRUE"'}
```

To enable global catalog LDAP searches to work with ONTAP for Windows 2008 R2 and later, you should modify the following UNIX attributes to replicate across the Global Catalog servers:

- gecos
- gidNumber
- memberUid
- nisMapName (if using netgroups)
- nisMapEntry (if using netgroups)
- nisNetgroupTriple (if using netgroups)
- uid
- uidNumber
- unixHomeDirectory
- unixUserPassword

You can verify which UNIX LDAP attributes have the value set to TRUE with this PowerShell command:

```powershell
PS C:\> Get-ADObject -SearchBase "cn=Schema,cn=Configuration,dc=ntap,dc=local" -LDAPFilter 
"(isMemberOfPartialAttributeSet=TRUE)" -Properties LDAPDisplayName | Select LDAPDisplayName | 
findstr -i "member uid gid unix"
```

Keep in mind the following caveats:
The use of global catalog servers for searches can add significant load and traffic to those servers. If you use the global catalog for LDAP searches, be sure that enough servers are available to handle the load.

Modifying the Active Directory schema can be very dangerous. Modify it with caution and document all changes in extreme detail. If possible, engage Microsoft support for assistance.

After you make changes to replicate attributes to the global catalog, you can either wait for the 15-minute replication window or force replication by using Active Directory Sites and Services.

Enabling global catalog searches in ONTAP LDAP clients is as simple as changing the LDAP port (-port) to 3268.

For example:

```bash
cluster::> ldap client modify -vserver DEMO -client-config DEMO -port 3268
vserver client-config port
--------- -------------- ----
DEMO DEMO          3268
```

In this example, before the Active Directory global catalog was modified to replicate UNIX attributes, lookups failed.

```bash
cluster::> getxxbyyyy getpwbyname -node node1 -vserver DEMO -username prof1
(vserver services name-service getxxbyyyy getpwbyname)
Error: command failed: Failed to resolve prof1. Reason: Entry not found for "username: prof1".
```

After the changes, lookups were successful.

```bash
cluster::> getxxbyyyy getpwbyname -node node1 -vserver DEMO -username prof1
(vserver services name-service getxxbyyyy getpwbyname)
pw_name: prof1
pw_passwd: 
pw_uid: 1100
pw_gid: 1101
pw_gecos: Professor
pw_dir: /home/prof1
pw_shell: /bin/sh
```

Use of LDAP to Serve Name Mapping Rules

In addition to UNIX users, groups, and netgroups, you can also use LDAP to query for name mappings in lieu of creating static name mapping entries on the ONTAP SVM. For consistency across UNIX and Windows users as they read, write, and navigate permissions, name mappings are necessary in multiprotocol NAS environments.

There are two main name mapping rule concepts:

- Symmetric name mapping is implicit name mapping between UNIX and Windows users who leverage the same user name; for example, Windows user DOMAIN\justin maps to UNIX user justin.
- Asymmetric name mapping is name mapping between UNIX and Windows users who leverage different user names; for example, Windows user DOMAIN\justin maps to UNIX user nfsdudeabides.

ONTAP natively supports symmetric name mappings with no need for name mapping rules and can support asymmetric name mappings with the name map ns-switch database.

Order of Operations for Name Mappings in ONTAP

When a user attempts to authenticate to a NAS mount or share, ONTAP uses a specific order of name mapping mechanisms to look for valid users or name map entries. This order ultimately depends on the first name service database value that is specified for the name map value in `vserver services`
name-service ns-switch. In the following example, ONTAP tries local files first and then LDAP.

Local files for name map values means the entries in the SVM’s name mapping table in vserver name-mapping.

```
cluster::> vservice services name-service ns-switch show -vserver DEMO -database namemap

Vserver: DEMO
Name Service Switch Database: namemap
Name Service Source Order: files, ldap
```

When you use LDAP for name mapping, ONTAP uses whatever the LDAP server is configured to use. In most cases, it is a symmetric name mapping, but it is also possible to use asymmetric values.

**Note:** Specify an external service in the name map database only if one is actually being used for asymmetric name mappings. If you specify a server that does not have any name mapping rules configured, it adds latency to requests and creates slow authentication or failures.

If no name mapping can be found in the name services entries for the user, then ONTAP tries to fall back on the default values that are set for the NFS or CIFS/SMB server. The use of this value depends on the protocol that is attempting access, the volume security style, and the name mapping direction that is requested. Table 9 shows the differences.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Security Style</th>
<th>Name Mapping Direction</th>
<th>Default User</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFS</td>
<td>UNIX</td>
<td>n/a (UID lookup only)</td>
<td>n/a</td>
</tr>
<tr>
<td>NFS</td>
<td>NTFS</td>
<td>UNIX -&gt; Windows</td>
<td>Default Windows user (NFS option default-win-user)</td>
</tr>
<tr>
<td>CIFS/SMB</td>
<td>UNIX</td>
<td>Windows -&gt; UNIX</td>
<td>Default UNIX user (CIFS option default-unix-user; pcuser by default)</td>
</tr>
<tr>
<td>CIFS/SMB</td>
<td>NTFS</td>
<td>Windows -&gt; UNIX (initial authentication)</td>
<td>NTFS ACLs are used after initial entry. Default UNIX user (CIFS option default-unix-user; pcuser by default)</td>
</tr>
</tbody>
</table>

**Asymmetric Name Mapping from Windows to UNIX Users in LDAP**

If your environment relies on bidirectional asymmetric name mapping from LDAP in ONTAP, create name mapping rules per SVM for the Windows-UNIX name mappings. However, there is a limit of 1,024 rules per SVM. If you need more rules than are allowed by the cluster, then you must modify the LDAP server attributes to include a UNIX user name with the same value as the Windows user name. Clients still pick up the desired UID/GID in this case.

You can use the LDAP client schema options in Table 10 to configure LDAP to serve name mappings.

<table>
<thead>
<tr>
<th>New LDAP Schema Attribute</th>
<th>What It Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>-windows-to-unix-object-class</td>
<td>Provides the LDAP attribute to define the Windows-to-UNIX name mapping object class. Object classes are used to group multiple LDAP objects to enable faster searches. The default value in AD-IDMU is User. For RFC 2037 schemas, the value is set to posixAccount.</td>
</tr>
<tr>
<td>-windows-to-unix-attribute</td>
<td>Provides the LDAP attribute for the value that is used for mapping a Windows user to a UNIX user. The default value</td>
</tr>
</tbody>
</table>
New LDAP Schema Attribute | What It Does
--- | ---
for AD-IDMU schemas in ONTAP is sAMAccountName. For RFC 2307 schemas, the value defaults to windowsAccount.

- `windows-to-unix-no-domain-prefix`
  This option controls whether the attribute value in `windows-to-unix-attribute` has the domain prefix added to it. (The default is false.) Because sAMAccountName is represented by a single user name (rather than DOMAIN\username) and because msDS-PrincipalName is not a value that can be used in LDAP search, domain prefixes might be necessary to enable functional asymmetric name mapping. The need for this value depends on the LDAP schema and attributes that are being used, as well as whether multiple domain name mappings are present for multiple unique Windows domains.

These options allow bidirectional asymmetric name mappings from both Windows to UNIX and UNIX to Windows from LDAP servers. The attribute values for these options depend on what your environment looks like.

For most Active Directory LDAP servers, the values for asymmetric name mappings are as follows:

- `windows-to-unix-object-class User`
- `windows-account-attribute sAMAccountName`
- `windows-to-unix-attribute sAMAccountName`
- `windows-to-unix-no-domain-prefix true`

With the preceding values, Active Directory LDAP works with name mappings out of the box. Any variations on the default schemas must be accounted for.

To set a user name mapping in LDAP on a user, simply populate the `uid` field in the attributes with the alternately named user. Note the difference in Figure 15 in user names for `uid` and for `sAMAccountName`.

**Figure 15) Use of LDAP for asymmetric name mapping.**

The following example shows a UNIX user named oracle mapping to a Windows user named oracle2 and vice versa.

**Example of working UNIX -> Windows and Windows -> Unix name mappings**

```
cluster::*> access-check name-mapping show -vserver DEMO -direction win-unix -name oracle
'oracle2' maps to 'oracle'
cluster::*> access-check name-mapping show -vserver DEMO -direction unix-win -name oracle2
'oracle2' maps to 'NTAP\oracle2'
```

**Asymmetric Name Mapping of UNIX Users to Windows Users**

The LDAP schema that is defined in ONTAP contains an attribute called “ONTAP Name Mapping windowsAccount Attribute” (`-windows-account-attribute`) that defines which LDAP schema
attribute to use when mapping UNIX names to Windows names. The default value of this attribute is sAMAccountName, which is the standard field that is used for Windows accounts when new users are created. This value can be modified if necessary, by creating a custom LDAP client schema, as covered in the section "Create Custom LDAP Schemas."

**UNIX-to-Windows Name Mapping Across Multiple Domains**

In some scenarios, UNIX-to-Windows name mapping might need to consider multiple users in different domains.

For example:

- The DEMO SVM has a CIFS/SMB server in NTAP.LOCAL, which is also using LDAP for UNIX users.
- DEMO’s LDAP client points to NTAP.LOCAL for UNIX user lookup.
- DEMO acquires a new company, Company B, and creates a new SVM called COMPANYB.
- Company B’s Active Directory domain CORE-TME.NETAPP.COM also has UNIX users.
- NTAP.LOCAL and CORE-TME.NETAPP.COM have a bidirectional trust created.
- This trust means that both domains can query Windows users in the other’s domain.

The goal here is to allow ONTAP to perform name mappings from UNIX users to Windows users, regardless of which domain they reside in. Because the domains are trusted, both SVMs can find the other domain’s Windows users with no issue.

But what has to happen first is that both SVMs must be able to find the UNIX users in LDAP, regardless of where they live. With the initial configuration of LDAP clients, each SVM can find its own UNIX users, but they can't find the trusted domain’s UNIX users.

To get multiple domains to map properly in both directions, you must perform a few steps:

1. Configure the LDAP clients to search multiple unique LDAP servers.
2. Configure UNIX-to-Windows mappings across multiple domains.

**Step 1: Configure the LDAP Clients to Search Multiple Unique LDAP Servers**

For both SVMs to be able to find users in LDAP, you have two options with Active Directory:

- Configure Active Directory to replicate UNIX attributes to the global catalog (as per the “Active Directory Global Catalog Searches” section).
- Leverage LDAP referrals (as per the section 3.6, “LDAP Referrals (Chase Referrals”)).

Each option has a few requirements involved.

**Global Catalog LDAP Searches for UNIX Identities**

For LDAP searches that use the global catalog to work across multiple domains, you need the following:

- Multiple domains in the same forest:
  - Multiple domains in different forests can’t access the global catalog.
- Domain trusts that work properly.
- UNIX attributes replicated to the forest level of the global catalog.
- The ONTAP SVM LDAP client configured for the Active Directory domain of the forest.
- DNS that functions properly.
- Time in sync across the domains.
- The LDAP client set to port 3268 (port 3269 is not supported by ONTAP).
- Bind user that can view objects in both domains.

**LDAP Referrals for UNIX Identities**

You can use LDAP referrals when domains are in different forests or when the global catalog can’t be used. You can also use LDAP referrals for LDAP servers that are not Windows Active Directory based. For LDAP referrals to work properly, the following must be true:

- LDAP referrals are enabled on the SVM LDAP client (-referrals-enabled).
- LDAP port 389 is used (port 636 is not supported for use with LDAP referrals).
- The LDAP server can support referrals (also known as chase referrals).
- The LDAP server list (-servers) is either multiple LDAP servers (IP addresses or hosts) or a fully qualified domain name (FQDN) with load-balanced IP addresses, or the Active Directory domain (-ad-domain) is set and DNS is configured with multiple SRV records for the LDAP servers.
- Multiple DNs are configured in base DN (-base-dn) and optionally in user, group, and netgroup DNs (-user-dn, -group-dn, -netgroup-dn).
- The bind level (-min-bind-level) and bind DN (-bind-dn) can bind to all LDAP servers listed.

In the following example, the LDAP client was set to use LDAP referrals to query LDAP servers in both the `NTAP.LOCAL` domain and the `CORE-TME.NETAPP.LOCAL` domain, which are trusted domains that are not in the same forest.

**LDAP Client Configuration Example for Multiple Domains with LDAP Referrals**

In this example, the fields that are needed for LDAP referral chasing to work are highlighted in yellow.

```
cluster::> ldap client show -client-config LDAP

Vserver: COMPANYB
Client Configuration Name: LDAP
LDAP Server List: =
(DEPRECATED)-LDAP Server List: =
 Active Directory Domain: ntap.local
Preferred Active Directory Servers: =
```

© 2020 NetApp, Inc. All Rights Reserved.
Bind Using the Vserver's CIFS Credentials: true
Schema Template: DEMO2
LDAP Server Port: 389
Query Timeout (sec): 3
Minimum Bind Authentication Level: sasl
Bind DN (User): -
Base DN: DC=CORE-TME,DC=NETAPP,DC=COM;DC=NTAP,DC=LOCAL
Base Search Scope: subtree
Vserver Owns Configuration: true
Use start-tls Over LDAP Connections: false
Client Session Security: none
LDAP Referral Chasing: true

In the preceding example, the Active Directory domain is NTAP.LOCAL. DNS SRV LDAP records have been added to the DNS server for the CORE-TME domain, the CORE-TME zone has been added as a secondary zone to the DNS server, and ONTAP can query both from the SVM.

With the preceding settings, the UNIX users in both domains can now be queried.

UNIX User and Group ID Collisions
In some cases, identical user names or numeric IDs might be in multiple domains. This situation can create problems for ONTAP, because it does not know which UID, user name, or GID is the correct one. Instead, it simply returns the first one that it finds. If you plan to use multiple domains for LDAP lookups, to avoid inconsistency in user and group lookups, make sure that there are no duplicate users or groups across the domains.

Step 2: Configure UNIX-to-Windows Mappings Across Multiple Domains
In some cases, a user might exist only as a UNIX user in one LDAP client configuration (such as UNIX LDAP or Centrify) and must be able to map to a valid Windows user in another domain.

In the following example, a user was created in CORE-TME named user, and the UNIX user name was set to username. There are no asymmetric name mapping rules, so username tries to find a Windows user named CORE-TME\username by default. It finds the UNIX UID but fails to map to a valid Windows user, even though a valid Windows user named NTAP\username is in the NTAP.LOCAL domain that the SVM can see properly—and that maps Windows to UNIX properly.
Vserver: COMPANYB (internal ID: 3)

Error: Get user credentials procedure failed
[  0 ms] Determined UNIX id 1998 is UNIX user 'username'
[  1] Trying to map 'username' to Windows user 'username' using
[  2] Implicit mapping
[  3] Using a cached connection to
stme-infra02.core-tme.netapp.com
[  4] Could not find Windows name 'username'
** [  4] FAILURE: Name mapping for UNIX user 'username' failed. No
** mapping found

Error: command failed: Failed to get user credentials. Reason: "SecD Error: Name mapping does not
exist".

cluster::*> access-check authentication show-creds -vserver COMPANYB -win-name NTAP\username -
list-name true -list-id true
(vserver services access-check authentication show-creds)

(NTAP\username (Windows Domain User))

GID: 513 (Domain Users)
Supplementary GIDs:
  513 (Domain Users)

Primary Group SID: S-1-5-21-3552729481-4032800560-2279794651-513 NTAP\DomainUsers (Windows
Domain group)

Windows Membership:
  S-1-5-21-3552729481-4032800560-2279794651-513 NTAP\DomainUsers (Windows Domain group)
  S-1-5-21-0-0-0-497 NT AUTHORITY\Claims Valid (Windows Well known group)
User is also a member of Everyone, Authenticated Users, and Network Users

When a UNIX user lives in the same trusted domain as the Windows user, the UNIX-to-Windows name
mapping works without the need to configure anything. For example, if the LDAP configuration points to
CORE-TME and a user named prof1 lives in NTAP.LOCAL with its corresponding Windows user, things
work as expected.

cluster::*> access-check authentication show-creds -vserver COMPANYB -unix-user-name prof1 -list-
name true -list-id true
(vserver services access-check authentication show-creds)

UNIX UID: 1100 (prof1) <> Windows User: S-1-5-21-3552729481-4032800560-2279794651-1110
(NTAP\prof1 (Windows Domain User))

GID: 1101 (ProfGroup)
Supplementary GIDs:
  1101 (ProfGroup)
  10000 (Domain Users)
  1201 (group1)
  1202 (group2)
  1203 (group3)
  1220 (sharedgroup)

Primary Group SID: S-1-5-21-3552729481-4032800560-2279794651-1111 NTAP\ProfGroup (Windows
Domain group)

Windows Membership:
  S-1-5-21-3552729481-4032800560-2279794651-1106 NTAP\group2 (Windows Domain group)
  S-1-5-21-3552729481-4032800560-2279794651-513 NTAP\DomainUsers (Windows Domain group)
User is also a member of Everyone, Authenticated Users, and Network Users

To get the UNIX-to-Windows mappings to work properly across domains in this scenario, you must
perform two steps:
• Create a UNIX-to-Windows name mapping rule that maps all UNIX users to wildcards for the domain and user.
• Add a name-mapping-search entry to the SVM to tell ONTAP to look in other trusted domains for UNIX-to-Windows name mappings.

This example does the following:

```
cluster::*> vserver name-mapping create -vserver COMPANYB -direction unix-win -pattern * - replacement '\\' -position 1
cluster::*> name-mapping-search add -vserver COMPANYB -trusted-domains NTAP.LOCAL
```

When that step is done, the previously nonworking username UNIX user now maps to the Windows user NTAP\username.

```
cluster::*> access-check authentication show-creds -vserver COMPANYB -unix-user-name username - list-name true -list-id true
  (vserver services access-check authentication show-creds)
  (NTAP\username (Windows Domain User))
  GID: 513 (Domain Users)
  Supplementary GIDs:
    513 (Domain Users)
  Primary Group SID: S-1-5-21-3552729481-4032800560-2279794651-513 NTAP\DomainUsers (Windows Domain group)
  Windows Membership:
    S-1-5-21-3552729481-4032800560-2279794651-513 NTAP\DomainUsers (Windows Domain group)
    S-1-18-2 Service asserted identity (Windows Well known group)
    S-1-5-21-0-0-0-497 NT AUTHORITY\Claims Valid (Windows Well known group)
  User is also a member of Everyone, Authenticated Users, and Network Users
```

### Active Directory Lightweight Directory Services

Rather than setting up an entire domain controller, you might want to provide LDAP services for UNIX users and groups with a standalone Windows server. For example, if you want a place to serve users and groups but don’t need Kerberos authentication, or if you need just LDAP functionality for an application, you can use Lightweight Directory Services (LDS). This directory service is available through the Active Directory LDS feature.

To manage users, groups, and netgroups in LDS, you must use ADSI Edit. Utilities such as Active Directory Users and Computers are intended for domain user management and do not work with standalone LDS instances.

To set up Active Directory LDS for use with UNIX identities, go to this link:

[Active Directory LDS Identity Mapping for Services for NFS](https://support.netapp.com/)

To use Active Directory LDS with ONTAP, follow the same configuration steps as you would with any other LDAP server. The following example shows a query from an ONTAP SVM to a standalone Active Directory LDS server that is running on Windows 2019.

```
cluster::*> getxxbyyy getpwbyname -node node1 -vserver NFS -username lds -show-source true
  (vserver services name-service getxxbyyy getpwbyname)
Source used for lookup: LDAP
pw_name: lds
pw_passwd:
pw_uid: 1001
pw_gid: 1101
pw_gecos:
pw_dir:
```

[56] How to Configure LDAP in ONTAP © 2020 NetApp, Inc. All Rights Reserved.
4.4 Configure Secure LDAP

ONTAP provides several methods to secure LDAP communication over the wire, with end-to-end encryption that uses industry-standard security mechanisms such as TLS 1.2 and AES-256 Kerberos. The following section covers how to configure these methods at a high level. For more detailed steps, consult your LDAP server vendor's documentation.

StartTLS and LDAPS

In this report, the section “StartTLS (Transport Layer Security) Versus LDAP Over SSL (LDAPS)” highlights the differences between StartTLS and LDAPS. Configuration for both of these is very similar: Both require certificates to be present on the LDAP server and on the ONTAP SVM. This part of the setup is generally the most complicated.

The ONTAP LDAP client configuration is relatively simple. For LDAPS, set the port to 636; ONTAP then knows to use LDAP S for binding and queries. To use StartTLS, set the port to 389 and enable the -use-start-tls option. ONTAP takes care of the rest, provided that the certificates have been installed on the SVM.

There are three general steps to configure LDAPS or StartTLS:

1. Configure certificate services on the LDAP server.
2. Install the certificates in the desired SVM by using security certificate commands.
3. Configure the LDAP client to use StartTLS or LDAPS.

The following sections cover these steps in further detail.

Manage Certificates in FreeIPA

Use the following links to create certificates in FreeIPA for use with LDAP:

- FreeIPA: Howto/Client Certificate Authentication with LDAP
- Red Hat: Managing Certificates for Users, Hosts, and Services

Manage Certificates in Windows Active Directory LDAP

To manage certificates with Windows Active Directory, you must have the Certificates feature installed and configured in your environment. This feature enables you to generate the necessary certificates to use with ONTAP for secure LDAP that uses SSL or StartTLS. Use the Microsoft procedures or contact Microsoft for assistance.

After you have installed the Certificates feature, to manage certificates:

1. Go to the Manage Computer Certificates window. Right-click Personal > Certificates and select All Tasks > Request New Certificate.
2. Follow the Certificate Enrollment wizard and select Domain Controller Authentication as your enrollment policy.

3. Click Enroll, and the certificate is ready to be exported.
4. Navigate to the new certificate in the Personal -> Certificates folder. It should have the same name as the Active Directory domain. Right-click and select All Tasks -> Export.

![Certificate Export wizard](image)

5. Use the Certificate Export wizard to create a new .cer file. You will use this file to install the certificate into the ONTAP SVM for LDAP over SSL or StartTLS. Base-64 .cer gives you a file that you can use immediately; DER encoded binary must be converted by using the certutil in Windows.

![Certificate Export Wizard](image)

6. When you have completed the wizard, you can open the .cer file. You should see a long text string that starts with:

```
-----BEGIN CERTIFICATE-----
```

7. Now you are ready to install the certificate in the ONTAP SVM.

**Install Certificates in ONTAP**

SVMs in ONTAP provide a method to import security certificates from LDAP servers through the security certificate commands.

```
cluster:/> security certificate ?
ca-issued> Show Digital Certificates Issued by Self-Signed CA
config> The config directory
create> Create and Install a Self-Signed Digital Certificate
delete> Delete an Installed Digital Certificate
```
By default, the admin SVM contains some well-known CA certificates (such as Verisign and Amazon), which you can copy and use in other SVMs if necessary. You can use the `security certificate install` command with any certificate type, whether it is Active Directory LDAP or not.

The previous section showed how to generate a self-signed certificate for use with Active Directory LDAP. To use that certificate, run the following command to install a `server-ca` certificate:

```
cluster::*> security certificate install -type server-ca -vserver DEMO -cert-name MS-LDAP
```

After that command has been run, it asks you to copy and paste the contents of your `.cer` file. When that step is finished, hit Enter and then the certificate is installed.

You should keep a copy of the CA-signed digital certificate for future reference.

```
The installed certificate's CA and serial number for reference:
CA: NTAP-ONEWAY-CA
Serial: 5500000004B5165DB56662E5E00000000004
```

You can view the certificate with the following:

```
cluster::*> security certificate show -vserver DEMO -common-name MS-LDAP
```

Now you can test it out.

**Configure and Test LDAPS or StartTLS**

After you install your `server-ca` certificate from your LDAP server, you can enable either StartTLS or LDAPS for use with LDAP binds and queries. If something was incorrect in the certificate configuration or installation, ONTAP prevents configuration changes or LDAP client creation if it can’t do a basic bind by using the specified security settings. For example, when the proper certificate has not been configured, the following errors result:

```
cluster::*> ldap client modify -client-config DEMO -vserver DEMO -use-start-tls true
```

Error: Validate the Ldap configuration procedure failed

```
[  6 ms] Hostname found in Name Service Cache
[    7] Successfully connected to ip 10.193.67.236, port 389 using TCP
[   13] Unable to start TLS: Server is unavailable
[   13] Unable to connect to LDAP (NIS & Name Mapping) service on oneay.ntap.local
**[   13] FAILURE: Unable to make a connection (LDAP (NIS & Name
```

© 2020 NetApp, Inc. All Rights Reserved.
** Error: command failed: The LDAP client configuration "DEMO" for Vservers "DEMO" is an invalid configuration.

cluster::*> ldap client modify -client-config DEMO -vserver DEMO -port 636

Error: Validate the Ldap configuration procedure failed
| 4 ms | Hostname found in Name Service Cache
| 10 | Successfully connected to ip 10.193.67.236, port 636 using TCP
| 67 | Unable to start LDAPS: Can't contact LDAP server
| 67 | Unable to connect to LDAP (NIS & Name Mapping) service on oneway.ntap.local (Error: Can't contact LDAP server)
| 67 | No servers available for LDAP_NIS_AND_NAME_MAPPING, vserver: 10, domain: **
**| 67 | FAILURE: Unable to make a connection (LDAP (NIS & Name Mapping)): result: 6940
** Error: command failed: The LDAP client configuration "DEMO" for Vservers "DEMO" is an invalid configuration.

This approach is a safeguard against outages caused by incorrect LDAP configuration. You can bypass this check with the -skip-config-validation option.

A sign that LDAP over SSL or StartTLS is working is the mere fact that the create or modify command works without error.

cluster::*> ldap client modify -client-config DEMO -vserver DEMO -use-start-tls true

Warning: You may also want to modify "-use-start-tls-for-ad-ldap" option to "true" using "vserver cifs security modify" command for the following Vserver(s): DEMO.

You can perform additional checks with the commands in the section "ONTAP CLI Commands for LDAP Troubleshooting."

Figure 16 and Figure 17 are examples of packet captures from identical LDAP queries that use StartTLS and LDAPS. The communication is nearly identical, except StartTLS has a TLS handshake and travels over port 389.

**Figure 16** Packet capture of LDAP StartTLS.

**Figure 17** Packet capture of LDAPS.

**LDAP Signing and Sealing (LDAP Session Security)**

Windows Active Directory LDAP provides a way to secure LDAP communication natively, with no need to configure security certificates. The LDAP client option that controls this method is -session-security.
If you use LDAP signing and sealing, you should also consider configuring the CIFS/SMB security settings as described in the section “CIFS/SMB Server Security Considerations.”

You have three options to configure session security:

- **None.** No session security is enforced.
- **Sign.** Session security is enforced for signing (integrity verification) LDAP sessions only. In this setup, LDAP queries can be seen over the wire in packet captures.
- **Seal.** Session security is enforced for both signing and sealing LDAP sessions. In this configuration, LDAP queries are encrypted and cannot be seen in packet captures.

Naturally, signing and sealing is the most secure, but it also incurs the most overhead for processing and can add some latency to queries. Results depend on the network load, the server load, and the size of queries. NetApp encourages you to carry out testing.

Figure 18 and Figure 19 show packet captures of LDAP signing and sealing, and the differences in what you can and cannot see in packet captures.

Figure 18) Packet capture of LDAP with signing enabled.
Binding as the CIFS/SMB Server

When you use Windows Active Directory LDAP for UNIX user and group lookups and the same SVM also has a CIFS/SMB server configured, you can use the CIFS/SMB server machine account to bind to LDAP, rather than using a user name/password or anonymous binds. Binding as a CIFS/SMB server is simple:

Set the option `bind-as-cifs-server` to `true` in the LDAP client and confirm that the configured CIFS/SMB server has the appropriate DNS entries. ONTAP sets the Kerberos SPN for the CIFS/SMB server automatically when the CIFS/SMB server is created.

Binding as the CIFS/SMB machine account uses the same security mechanisms that the Active Directory domain supports. Machine account authentication tries to negotiate through Kerberos (by using GSS-SPNEGO) first. If a valid SPN exists on the KDC, then Kerberized binds are used with the strongest supported encryption type available on the KDC. For Windows 2008 and later, that encryption is AES-256. Figure 20 shows a packet capture of an LDAP bind using the CIFS/SMB server.
If Kerberos SPNs are not available, the machine account authentication falls back to NTLM. If NTLM is disallowed in the domain, then the ONTAP LDAP client checks to see whether a username/password is configured. Otherwise, binding fails.

### Considerations for Binding as a CIFS/SMB Server

When you bind as a CIFS/SMB server, consider the following:

- Ensure that CIFS/SMB is licensed and that a CIFS/SMB server is configured.
- Ensure that DNS entries exist for the CIFS server name that is being used.
- Ensure that DNS is configured on the SVM in ONTAP.
- In the LDAP client, ensure that the Active Directory domain is configured with the `-ad-domain` option.
- Set the minimum bind level (`-min-bind-level`) on the LDAP client to SASL.
- If NTLM is disabled in your domain, consider setting a bind user and password for fault tolerance in case Kerberos binds fail.

### CIFS/SMB Server Security Considerations

When you use Active Directory for LDAP services, there are also some CIFS/SMB specific options that you can configure to further secure LDAP traffic.

The following options are available in the advanced privilege:

```bash
cluster::*> cifs security modify -vserver DEMO ?
   [ -kerberos-clock-skew <integer> ]
   Maximum Allowed Kerberos Clock Skew
   [ -kerberos-ticket-age <integer> ]
   Kerberos Ticket Lifetime
   [ -kerberos-renew-age <integer> ]
   Maximum Kerberos Ticket Renewal Days
   [ -kerberos-kdc-timeout {1..23} ]
   Timeout for Kerberos KDC Connections (Secs)
   [ -is-signing-required {true|false} ]
   Require Signing for Incoming CIFS Traffic
```
4.5 Use of LDAP to Host Netgroups

You can leverage netgroup functionality in LDAP, which is not possible in NIS. Netgroups enable storage administrators to control access to a series of hosts by using a group, rather than having to create several different rules per host. In LDAP, you can use ONTAP to store and to query host names, IP addresses, and netgroup entries. The use of LDAP as a NIS server is covered in RFC 2307. Currently, only host names and IP addresses are supported for use with netgroups in ONTAP.

About NIS Objects and Attributes in LDAP

NIS object types in LDAP are determined by way of the objectClass attribute. The objectClass attribute that is set on an object determines how ONTAP and other LDAP clients query LDAP for netgroup-related objects. For netgroups, the nisNetgroup object class is used by default in most schemas, including Active Directory and FreeIPA. Table 11 presents a summary.

Table 11) Object class types for netgroups.

<table>
<thead>
<tr>
<th>objectClass</th>
<th>Used For</th>
<th>Common Attributes Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>nisMap</td>
<td>NIS maps</td>
<td>nisMapName</td>
</tr>
</tbody>
</table>
| nisNetgroup       | Netgroups         | nisMapName  
nisNetgroupTriple |
| nisObject         | Netgroups         | nisMapEntry  
nisMapName |
|                   | netgroup.byhost entries |                   |

NIS Object Terminology

Table 12 defines terminology for specific aspects of NIS objects.

The CIFS/SMB security options apply to LDAP lookups for CIFS/SMB communication. For UNIX LDAP communication with Active Directory, use the LDAP client settings. To get the best possible security, use both the CIFS/SMB and LDAP client security settings.
Table 12) NIS object terminology.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIS map</td>
<td>NIS maps are designed to centralize and to replace commonly found files in the /etc directory of Linux and UNIX clients.</td>
</tr>
<tr>
<td></td>
<td>ONTAP currently supports the following NIS map types:</td>
</tr>
<tr>
<td></td>
<td><code>passwd.byname</code> and <code>passwd.byuid</code></td>
</tr>
<tr>
<td></td>
<td><code>Group.byname</code> and <code>group.bygid</code></td>
</tr>
<tr>
<td></td>
<td><code>Netgroup</code></td>
</tr>
<tr>
<td></td>
<td><code>netgroup.byhost</code> (as of ONTAP 8.3)</td>
</tr>
<tr>
<td></td>
<td>ONTAP does not currently support host name resolution in NIS.</td>
</tr>
<tr>
<td></td>
<td>For more information about NIS maps, see Oracle NIS Maps.</td>
</tr>
<tr>
<td>Netgroup</td>
<td>A netgroup is a set of (host,user,domain) triples that are used for permission and export access checking.</td>
</tr>
<tr>
<td></td>
<td>ONTAP currently supports only hosts in netgroup entries.</td>
</tr>
<tr>
<td></td>
<td>The netgroup must use only a comma (,) as the delimiter.</td>
</tr>
<tr>
<td></td>
<td>For more information about netgroups, see the Linux manual pages and the FreeBSD Manual Pages.</td>
</tr>
<tr>
<td>Triple</td>
<td>A netgroup triple refers to the series of entries in a netgroup file, consisting of (host,user,domain).</td>
</tr>
<tr>
<td></td>
<td>A valid triple in ONTAP consists of (host,,). When you designate a blank field, be sure to follow the netgroup file standard guidance for</td>
</tr>
<tr>
<td></td>
<td>your operating system. Special characters, such as dashes, can cause lookups to fail and access to be denied. Host names that are</td>
</tr>
<tr>
<td></td>
<td>used in netgroup triples require DNS resolution in ONTAP. For best results in netgroup translation, see the name services best practices</td>
</tr>
<tr>
<td></td>
<td>in TR-4067 and TR-4668.</td>
</tr>
<tr>
<td>netgroup.byhost</td>
<td><code>netgroup.byhost</code> entries are used to accelerate netgroup lookups by querying the name service for the group membership by host rather</td>
</tr>
<tr>
<td></td>
<td>than querying the entire netgroup. For netgroups with many entries, this process can reduce lookup time drastically and improve</td>
</tr>
<tr>
<td></td>
<td>performance. For more information about <code>netgroup.byhost</code> support, see the section “netgroup.byhost Support.”</td>
</tr>
</tbody>
</table>

ONTAP Interaction with Active Directory LDAP for Netgroups

In the schemas that ONTAP provides, the following attributes control lookups for netgroups and their members:

```
-nis-netgroup-object-class
-nis-netgroup-triple-attribute
-member-nis-netgroup-attribute
-cn-netgroup-attribute
```

In ONTAP 8.3 and later, the following attributes are provided to support `netgroup.byhost`; for more information, see the “netgroup.byhost Support” section:

```
-nis-object-class
-nis-mapname-attribute
-nis-mapentry-attribute
```

You can modify LDAP client schemas to change the default attributes for netgroups. In most cases, the provided read-only templates work, but in some cases, you might have to copy a schema template to a new template and modify it. For more information about schemas, see the section in this document titled “LDAP Schemas.”
The Active Directory schema has the following schema attributes added by default in Windows 2012 and later (the default attributes that ONTAP uses are in bold):

<table>
<thead>
<tr>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>memberNisNetgroup</td>
</tr>
<tr>
<td>msSFU30Name</td>
</tr>
<tr>
<td>msSFU30NetgroupHostAtDomain</td>
</tr>
<tr>
<td>msSFU30NetgroupUserAtDomain</td>
</tr>
<tr>
<td>msSFU30NisDomain</td>
</tr>
<tr>
<td>NisMap</td>
</tr>
<tr>
<td>nisMapEntry</td>
</tr>
<tr>
<td>nisMapName</td>
</tr>
<tr>
<td>nisNetgroup</td>
</tr>
<tr>
<td>nisNetgroupTriple</td>
</tr>
<tr>
<td>nisObject</td>
</tr>
</tbody>
</table>

Creating Netgroups With Active Directory LDAP

Active Directory netgroups can be controlled by using the utilities nis2ad and nismap, PowerShell, or GUI tools such as ADSI Edit.

Use of nismap to Create NIS Objects

With nis2ad, you can migrate existing maps from NIS to Active Directory, or you can create NIS maps from a local file. This utility is included in the Identity Management for UNIX feature in Windows 2008 and later. However, you generally do not need it unless you create new NIS maps outside of the default netgroup NIS map that AD-IDMU creates.

The nismap command enables granular management of NIS maps in addition to what nis2ad provides.

The -e flag lists the netgroup/nismap entry. The format follows the same formats that are used in NIS netgroup files and that are covered in the Linux manual pages.

Note: The ipHostNumber attribute is used for lookups of the NIS host IP information for most LDAP servers and clients. ONTAP does not support this attribute.

Following is a sample nismap command:

```
C:\>nismap add -a americas -s USA -c C:\nisadd.txt -e "hosts (host1,,) (host2,)")" netgroup
Activity = Adding 'map = 'netgroup'... SUCCESS
Adding the object in Active Directory Domain Services.
Object = 'hosts'
Object class = 'NisNetgroup'
container = 'CN=netgroup,CN=americas,CN=DefaultMigrationContainer30,DC=americas,DC=win2k12,DC=netapp,DC=com'.
SUCCESS
adding NIS entries to AD
```

In the preceding example, the following occurred (see Figure 21):

- An object called hosts was created.
- The object class of NisNetgroup was applied to the object.
- The default container was
  'CN=netgroup,CN=americas,CN=DefaultMigrationContainer30,DC=americas,DC=win2k12,DC=netapp,DC=com'.

The netgroup DN will be used when configuring the LDAP client in ONTAP using the -netgroup-dn field to provide DN filtering for netgroup LDAP queries.
You can view the attributes for the object by double-clicking the attribute and selecting Attribute Editor; see Figure 22.

Figure 22) Netgroup properties in Active Directory LDAP.
Use of PowerShell to Create or to Manage New Netgroups

PowerShell also includes cmdlets that give administrators a way to use a more familiar tool to create and to manage netgroups.

Those cmdlets are as follows:

<table>
<thead>
<tr>
<th>Cmdlet</th>
</tr>
</thead>
<tbody>
<tr>
<td>New-NfsNetgroup</td>
</tr>
<tr>
<td>Get-NfsNetgroup</td>
</tr>
<tr>
<td>Set-NfsNetgroup</td>
</tr>
<tr>
<td>Remove-NfsNetgroup</td>
</tr>
</tbody>
</table>

For example, create a new netgroup called `powershell`, of which the host `centos7.ntap.local` is a member.

```powershell
C:\> New-NfsNetgroup -NetGroupName powershell -AddMember centos7.ntap.local -LdapNamingContext "OU=netgroups,DC=NTAP,DC=local" -LdapServer ntap.local
```

When you run the preceding command, two entries are created in the location that you specified: the netgroup called `powershell` and a host entry named `centos7.ntap.local`. See Figure 23.

![Figure 23) Entries created by New-NfsNetgroup in Active Directory.](image)

By default, those entries use the classic netgroup functionality in ONTAP, which queries a netgroup, fetches all hosts in the netgroup, and populates the netgroup cache.

To modify the host entry to use `netgroup.byhost` functionality, simply rename the object by using the `Rename-ADObject` PowerShell cmdlet. This step is necessary to change the host name to append `.*` to the end of the name so that ONTAP knows that the DNS query ends there.

For example:

```powershell
PS C:\> Rename-ADObject -Identity "CN=centos7.ntap.local,OU=netgroups,DC=NTAP,DC=local" -NewName centos7.ntap.local."
```

After that step, you can use `netgroup.byhost` functionality with netgroups that are created in Active Directory. For more information about `netgroup.byhost` functionality, see the section "netgroup.byhost Support" in this document.

Use of ADSI Edit to Create New NIS Objects

Another way to create NIS objects is to use the ADSI Edit tool in Active Directory. To use ADSI Edit, make sure that it is installed.

Note: ADSI Edit should be used with extreme caution, because serious damage to the Active Directory schema can occur if it is not used correctly. If you need help in using ADSI Edit, contact Microsoft technical support.

After ADSI Edit is installed, open the ADSI Edit console and connect to the default naming context path. See Figure 24.
Figure 24) Connecting to default naming context.

After you are connected, the entire Active Directory schema is shown. If a container for NIS objects does not already exist, it might make sense to create one for organizational purposes.

To create a container, right-click when the desired location is highlighted and select New -> Object. This step brings up a new dialog box that enables you to specify the object type (or object class). See Figure 25.

Figure 25) Creating a new object.

- **nisMap** enables you to create a `netgroup.byhost` map entry.
- **nisNetgroup** creates a classic netgroup.
- **nisObject** enables you to create a host entry to use for netgroup lookups.

For example, the LDAP server in Figure 26 has two entries in the “netgroups” OU.
In the preceding example, `netgroup1` is a regular netgroup. It uses the typical `nisNetgroupTriple` logic, such as (hostname,-,-). See Figure 27.

Figure 27) Entries in netgroup1.

Create a netgroup.byhost Entry

To create a netgroup.byhost entry, create a `nisObject` for the host name, with .* appended to the end of the name. This tells ONTAP that the host entry’s DNS name is finished and can be used for host name lookup. For example, a host named `centos7.ntap.local` is created as `centos7.ntap.local.*` when creating the netgroup.byhost entry.

Following is a working netgroup.byhost netgroup query in ONTAP, using Active Directory LDAP:

```
cluster::*> getxxbyyy netgrpcheck -node node01 -vserver DEMO -netgroup powershell -clientIP 10.193.67.225 -show-source true
  (vserver services name-service getxxbyyy netgrpcheck)
Success. Client 10.193.67.225 is member of netgroup powershell
```
Searched using NETGROUP_BYHOST
Source used for lookup: LDAP

The nisMap attribute tells the LDAP server and client that the entry is a type of map. For an explanation of NIS maps and to see examples of different types of NIS maps, go to NIS Maps. ONTAP supports only netgroup.byhost as a NIS map.

The examples in Figure 28 show a netgroup and a netgroup.byhost object in Active Directory, along with their associated attributes.

Figure 28) Netgroup and netgroup.byhost entries in Active Directory LDAP.

Manage Netgroup Objects with the GUI in Windows

After a netgroup object has been created with ADSI Edit or one of the CLI methods, you can manage the entries with the Active Directory Users and Computers (ADUC) MMC. Because there are safeguards in ADUC, that method is preferred over ADSI Edit. Everything in Figure 28 except the CN field can be modified in the GUI, including adding, changing, and removing netgroup clients or changing the nisMapEntry. To modify the CN field, use the Rename-ADObject command in PowerShell.

Creating Netgroups With FreeIPA LDAP

You can also use FreeIPA LDAP to create and to manage netgroups. The easiest method is through the provided GUI webpage. To connect to the FreeIPA web interface, simply navigate to http://freeipaserver in a web browser.

Add Netgroups in FreeIPA LDAP

The netgroup management GUI is in Identity -> Groups in the FreeIPA GUI. See Figure 29.
To create new netgroups, click Add. To edit existing netgroups (add or remove hosts or IP addresses), select the netgroup name.

To add hosts, you can use the Hosts tab under the same Identity part of the GUI. See Figure 30.

After you have created a netgroup, you can verify that it works through the ONTAP CLI.

```
cluster::*=> getxxbyyy netgrpcheck -node node1 -vserver NFS -netgroup ipa-netgroup -clientIP 10.193.67.222 -show-source true
 SUCCESS. Client 10.193.67.222 is member of netgroup ipa-netgroup
Searched using NETGROUP_BYNAME
Source used for lookup: LDAP
```

**Netgroup Caches in ONTAP**

ONTAP uses several caches to store information such as host names and netgroups locally. This method is faster than having to retrieve this information from an external source each time that it is required.

Export policies and rules control access to NFS exports. Each export policy contains rules, and each rule contains parameters to control client access. Some of these parameters require ONTAP to contact an external source such as DNS or NIS servers to resolve objects such as domain names, host names, or netgroups. Communications with external sources can have associated latency because of the load, network, and so on. To improve performance, ONTAP reduces the amount of time that it takes to resolve export policy rule objects by storing information locally in several caches.
One main disadvantage to the use of caches to store information locally is that if the information on the external name server was changed after ONTAP retrieved and stored it locally, the caches might contain outdated information. As a result, client access requests that should succeed could fail, and client access requests that should fail could succeed. To help avoid such issues, ONTAP flushes caches automatically after a certain period and provides commands that enable you to view and to manually flush some of the export policy caches.

For more information about commands to view and flush caches and to view and modify timeout values, see the section “ONTAP CLI Commands for LDAP Troubleshooting.”

**NIS Netgroup Strict (nfs.netgroup.strict)**

In NetApp Data ONTAP® operating in 7-Mode, the option `nfs.netgroup.strict` allowed control of whether a netgroup entry required the @ sign so that Data ONTAP recognized the netgroup as a netgroup.

ONTAP currently has no equivalent to the `nfs.netgroup.strict` option. All netgroups in export policy rules must be designated with the @ sign to be recognized as netgroups. If no @ sign is present, ONTAP treats the entry as a host name and attempts to resolve the name in DNS or local hosts.

**netgroup.byhost Support**

`netgroup.byhost` entries can vastly speed up netgroup entry lookup. With `netgroup.byhost`, the cluster can avoid having to query every entry in a netgroup for access and instead can fetch the netgroup by way of LDAP lookup per host. In large environments with netgroups that have many entries, this approach can drastically speed up the time for lookups and can avoid access issues due to timeouts on LDAP queries. Support for `netgroup.byhost` was added to ONTAP starting with version 8.3.

**netgroup.byhost Example**

The section Creating Netgroups With Active Directory LDAP showed you how to create a netgroup by using Windows Active Directory. One of the netgroups that was created is named `netgroup1`, and it is also a `nisMapEntry`. Therefore, it can act as a standalone netgroup (`-is-netgroup-byhost-enabled false`) or as a way for host-performed netgroup queries to map to a netgroup (`-is-netgroup-byhost-enabled true`).

The aforementioned section also showed examples of host entries that you can use with `netgroup.byhost`.

For example, let's say that you have a host set in the `netgroup.byhost` object. That host is your Free IPA server, `centos8-ipa`, with an IP address of 10.193.67.222. Note in Figure 31 that .* is at the end of the name to let ONTAP know that there are no more parts of the name to look up in DNS.
As Figure 32 shows, netgroup1 has only the following nisNetgroupTriple entries.

Therefore:

- **When netgroup.byhost support is enabled**, then only centos8-ipa has access to exports that have netgroup1 assigned, because it is the only host that is defined.
- **When netgroup.byhost support is disabled**, then centos8-ipa is denied access and the hosts in the nisNetgroupTriple field get access.

You can verify this information with the command `export-policy check-access`. (You can find more information about this command in the section “export-policy.”)

**Example of netgroup.byhost Disabled**
In this example, because netgroup.byhost is disabled, the centos8-ipa client does not have access to the volume with the netgroup export policy, but clients in the nisNetgroupTriple field (10.193.67.225 and xcp) do.

<table>
<thead>
<tr>
<th>Path</th>
<th>Policy</th>
<th>Owner</th>
<th>Owner Type</th>
<th>Index Access</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/netgrpvol</td>
<td>netgroup</td>
<td>netgrpvol</td>
<td>volume</td>
<td>0</td>
<td>denied</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When you view the netgroup cache, you can see that the successful clients used netgroup_byname (or netgrp_byname) to gain access. The failed access shows a source of none, which means that it is in the negative cache.

**Example of netgroup.byhost Enabled**

In this example, the cache is cleared, and netgroup.byhost support is enabled on the LDAP client. This action reverses the results; centos8-ipa has access and the other clients do not.

<table>
<thead>
<tr>
<th>Path</th>
<th>Policy</th>
<th>Owner</th>
<th>Owner Type</th>
<th>Index Access</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/netgrpvol</td>
<td>netgroup</td>
<td>netgrpvol</td>
<td>volume</td>
<td>1</td>
<td>read-write</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When you view the netgroup cache, you can see that the successful clients used netgroup_byname (or netgrp_byname) to gain access. The failed access shows a source of none, which means that it is in the negative cache.

**Example of netgroup.byhost Enabled**

In this example, the cache is cleared, and netgroup.byhost support is enabled on the LDAP client. This action reverses the results; centos8-ipa has access and the other clients do not.

<table>
<thead>
<tr>
<th>Path</th>
<th>Policy</th>
<th>Owner</th>
<th>Owner Type</th>
<th>Index Access</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/netgrpvol</td>
<td>netgroup</td>
<td>netgrpvol</td>
<td>volume</td>
<td>1</td>
<td>read-write</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When you view the netgroup cache, you can see that the successful clients used netgroup_byname (or netgrp_byname) to gain access. The failed access shows a source of none, which means that it is in the negative cache.

**Example of netgroup.byhost Enabled**

In this example, the cache is cleared, and netgroup.byhost support is enabled on the LDAP client. This action reverses the results; centos8-ipa has access and the other clients do not.
Enable netgroup.byhost Support in ONTAP

netgroup.byhost support is not enabled by default in ONTAP. To enable it, the following options in the LDAP client configuration must be modified:

- `is-netgroup-byhost-enabled`
- `netgroup-byhost-dn`
- `netgroup-byhost-scope`

Naturally, `-is-netgroup-byhost-enabled` must be enabled to allow the use of `netgroup.byhost` functionality.

DN and scope specify the filters that you want for `netgroup.byhost` functionality. For more information, see the administration guides for your release of ONTAP. Keep in mind that support for this feature applies only to ONTAP 8.3 and later versions.

### 4.6 Configure External LDAP Clients

ONTAP SVMs are LDAP clients that can communicate with LDAP servers to query users and groups. You can also configure client OSs to use LDAP queries to the same LDAP server. With this configuration, ONTAP and Linux/NFS clients can use the same source for users and groups, which provides consistent access permissions with minimal file management.

This section covers the [SSSD LDAP module](https://www.netapp.com/products/clustered-data-storage-software/ontap/). Other LDAP clients are not covered in this section, but the same general concepts apply.

In general, you follow these steps:

1. Install or update the LDAP client if it is not already installed or updated.
2. Join the client to the Active Directory domain if you are using Active Directory LDAP and if you want to use Kerberos for LDAP binds.
3. Ensure that the DNS is configured properly on the client.
4. Configure the `nsswitch.conf` file to use `sss` for `passwd`, `group`, and `netgroup` (if necessary).
5. Configure the LDAP client to use LDAP for the id_provider.
6. Configure the LDAP client to use the LDAP client schema options that you want.

Detailed steps for SSSD are in the following section.

**Note:** For other LDAP client configurations, see the vendor documentation.

**SSSD—Linux LDAP Client**

**SSSD** is a system daemon that Red Hat and Fedora developed as a replacement for PADL, Samba Winbind, and other Active Directory–based PAM and nss modules. SSSD delivers access to different identity and authentication providers. To use the new LDAP interface, a new PAM module called `pam_sss` was created. SSSD includes an Active Directory provider type, enabling easy integration with Windows Active Directory 2003, 2008, and 2012. SSSD leverages TLS encryption and LDAP by using GSSAPI, which enables more secure LDAP binding and lookups over the wire.

The steps in this document cover setting up SSSD to use GSSAPI (Kerberos) for authenticated LDAP binds. SSSD uses the strongest Kerberos encryption type that is supported by the client and the Active Directory domain controller.

SSSD configuration uses the `/etc/sssd/sssd.conf` file on clients that support SSSD. Each time that you change the configuration, you should restart SSSD.

You can configure SSSD to cache the name database on the client. For performance reasons, NetApp recommends that you do this step. However, caching can cause confusion in troubleshooting, so when you restart the service during troubleshooting, use the following to clear the cache:

```
[client] # service sssd stop
[client] # rm -f /var/lib/sss/db/*
[client] # service sssd start
```

SSSD is also case sensitive by default. Because Microsoft Active Directory does not care about case sensitivity, NetApp recommends that you configure SSSD to ignore case sensitivity.

**RHEL, CentOS, and Fedora Client Configuration**

The following assumes that the kernel that is running supports the SSSD LDAP package. Some newer versions of Linux include SSSD by default in basic installations. If SSSD is not installed, install it.

To check for the application:

```
[client] # yum list | grep sssd
```

To install it:

```
[client] # yum install -y sssd
```

If the application is already installed, it might be beneficial to upgrade:

```
[client] # yum update -y sssd
```

For Active Directory LDAP environments, to make setup and use easier, join the Linux client to the domain. This step allows the client to use the KDC functionality of the Active Directory domain without extra setup steps, which provides secure binds through Kerberos and automatically configures SSSD and the `krb5.conf` file for Active Directory user lookup. UNIX identity lookups require a bit of additional configuration, but the brunt of the work is performed by the `realmd` application.

```
[client] # yum install -y realmd
```

Older clients might require manual Kerberos configuration or the use of `net ads` commands. For details, see TR-4616 and TR-4073.
After the applications are installed, you can start the configuration process.

**Configure DNS**

DNS is necessary for various functions, including Kerberos and LDAP. Specifically, DNS is used to resolve host names, which are used to query for SPNs. Also, DNS queries for LDAP SRV records.

You can configure DNS through one of the client-specific command utility wizards or through basic editing of the `/etc/resolv.conf` file. The DNS server and search domains that you apply should be able to query for host names and SPNs that are in use by the client.

For information about how to configure DNS in RHEL and CentOS, see [Changing Domain DNS Configuration](#) on the Red Hat Customer Portal.

**Join the Key Distribution Center**

Key Distribution Centers (KDCs) provide Kerberos authentication services to environments. LDAP applications such as SSSD use LDAP servers through logins called “binds.” Binds can be anonymous or use plaintext passwords, but, ideally, binds use as much security as possible. The use of Kerberos interaction with KDCs for binds is one way to promote such extensive security.

SSSD provides a way to bind to LDAP through a Kerberos SPN to encrypt binds with the strongest available Kerberos encryption type (currently AES-256 in newer KDCs). It can be done manually through the `sssd.conf` file, or it can be done automatically by using domain join tools, such as `realm`.

Following are the available `realm` commands:

```plaintext
# realm
realm discover -v [realm-name]
    Discover available realm

realm join -v [-U user] realm-name
    Enroll this machine in a realm

realm leave -v [-U user] [realm-name]
    Unenroll this machine from a realm

realm list
    List known realms

realm permit [-ax] [-R realm] user ...
    Permit user logins

realm deny --all [-R realm]
    Deny user logins
```

You can check that DNS is doing its job, as well as check for application dependencies, with the `realm discover -v [realm-name]` command.

```plaintext
# realm discover -v NTAP.LOCAL
    * Resolving: _ldap._tcp.ntap.local
    * Performing LDAP DSE lookup on: 10.193.67.236
    * Successfully discovered: NTAP.LOCALNTAP.LOCAL
    NTAP.LOCALNTAP.LOCAL
        type: kerberos
        realm-name: NTAP.LOCAL
        domain-name: NTAP.LOCALNTAP.LOCAL
        configured: no
        server-software: active-directory
        client-software: sssd
        required-package: sssd-tools
        required-package: sssd
        required-package: adcli
        required-package: samba-client
        ntap.local
            type: kerberos
```
Before you run the `realm join` command, check your `krb5.conf` file. In this example, you see that nothing is configured.

```
# cat /etc/krb5.conf
# Configuration snippets may be placed in this directory as well
includedir /etc/krb5.conf.d/

[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
# default_realm = EXAMPLE.COM

[realms]
# EXAMPLE.COM = {
#  kdc = kerberos.example.com
#  admin_server = kerberos.example.com
# }

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

Now join the KDC realm. If you are missing package dependencies, they are automatically installed.

```
# realm join NTAP.LOCAL
Password for Administrator:
* Installing necessary packages: adcli sssd-tools
```

Then you can check your `realm list`.

```
# realm list
NTAP.LOCAL
    type: kerberos
    realm-name: NTAP.LOCAL
    domain-name: ntap.local
    configured: kerberos-member
    server-software: active-directory
    client-software: sssd
    required-package: sssd-tools
    required-package: sssd
    required-package: adcli
    required-package: samba-client
    login-formats: %U@ntap.local
    login-policy: allow-realm-logins
```

When a realm is joined, the `krb5.conf` file is modified to include the new realm information.

```
default_realm = NTAP.LOCAL
[realms]
# EXAMPLE.COM = {
#  kdc = kerberos.example.com
#  admin_server = kerberos.example.com
# }

NTAP.LOCAL = {
}

[domain_realm]
# .example.com = EXAMPLE.COM
# example.com = EXAMPLE.COM
ntap.local = NTAP.LOCAL
.ntap.local = NTAP.LOCAL
```
And sssd.conf also gets the Active Directory information.

```plaintext
[domain/NTAP.LOCALNTAP.LOCAL]
ad_domain = NTAP.LOCALNTAP.LOCAL
krb5_realm = NTAP.LOCAL
realm_tags = manages-system joined-with-adcli
cache_credentials = True
id_provider = ad
krb5_store_password_if_offline = True
default_shell = /bin/bash
ldap_id_mapping = True
use_fully_qualified_names = True
fallback_homedir = /home/%u@%d
access_provider = ad
```

Configure the /etc/sssd/sssd.conf File

When you join a domain with realm, the SSSD configuration uses the id_provider of ad. This approach uses the SSSD UID/GID Algorithms (SSSD Active Directory Provider) method that is covered later in this document. The UNIX IDs that this method provides are not supported by ONTAP, so you must add a second configuration section in the SSSD configuration that includes LDAP as the id_provider.

The following is a sample configuration, which might vary depending on your environment:

```plaintext
[domain/DOMAIN]
auth_provider = krb5
chpass_provider = krb5
id_provider = ldap
ldap_search_base = dc=ntap,dc=local
ldap_schema = rfc2307bis
ldap_sasl_mech = GSSAPI
ldap_user_object_class = user
ldap_group_object_class = group
ldap_user_home_directory = unixHomeDirectory
ldap_user_principal = userPrincipalName
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true
ldap_user_search_base = cn=Users,dc=ntap,dc=local
ldap_group_search_base = cn=Users,dc=ntap,dc=local
ldap_sasl_authid = CENTOS7$@NTAP.LOCAL
krb5_server = ntap.local
krb5_realm = NTAP.LOCAL
krb5_kpasswd = ntap.local
use_fully_qualified_names = false
```

The preceding example uses:

- Kerberos binds through `ldap_sasl_mech` and `ldap_sasl_authid`
- The machine account as the `ldap_sasl_authid` SPN
- LDAP as the `id_provider`
- Short names for lookups (`use_fully_qualified_names = false`)

The `realm` command configured SSSD to use fully qualified names by default (for example, `user@REALM.COM`), so the LDAP lookups depend on how the name lookups are issued.

For example, if you look for `prof1@NTAP.LOCAL`, SSSD uses the Active Directory provider and generates a UID based on the Active Directory security identifier (SID).

```plaintext
# id prof1@NTAP.LOCAL
uid=1587401110(prof1@NTAP.LOCALNTAP.LOCAL) gid=1587400513(domainusers@NTAP.LOCALNTAP.LOCAL)
groups=1587400513(domainusers@NTAP.LOCALNTAP.LOCAL),1587401106(group2@NTAP.LOCALNTAP.LOCAL),1587401122(sharedgroup@NTAP.LOCALNTAP.LOCAL),1587401111(profgroup@NTAP.LOCALNTAP.LOCAL),1587401105(group1@NTAP.LOCALNTAP.LOCAL),1587401107(group3@NTAP.LOCALNTAP.LOCAL)
```
If you look for `prof1`, you get the UNIX attributes from LDAP rather than having SSSD generate a UID for you.

```
# id prof1
uid=1100(prof1) gid=1101(ProfGroup)
groups=1101(ProfGroup),1201(group1),1203(group3),1202(group2),1220(sharedgroup)
```

This configuration provides flexibility for environments that use both types of user lookups.

**SSSD UID/GID Algorithms (SSSD Active Directory Provider)**

SSSD, which is an LDAP client that is provided by Red Hat, offers two different ways to provide user and group identities. When you configure SSSD to use “LDAP” as the ID provider, SSSD performs normal LDAP search queries that use RFC 2307–based schema standards for user and group lookups. For instance, when a user named `user` is looked up by using the LDAP provider, SSSD searches for `uid=user` and for the pertinent information for user authentication, such as UID numbers, home directory paths, and group memberships.

SSSD also supports an [Active Directory integrated solution](#) (ID provider “AD”). This solution provides UNIX IDs based on an algorithm that uses the preexisting Windows SIDs for users and groups to create numeric UID and GID values for UNIX identity management. The intention is to eliminate the need to create UNIX attributes for each user and group, and instead to rely on SSSD to do the work.

For instance, on the following example SSSD client, SSSD is configured to use both `LDAP` and `AD` for ID providers:

```ini
[domain/DOMAIN]
auth_provider = krb5
chpass_provider = krb5
id_provider = ldap
ldap_search_base = dc=ntap,dc=local
ldap_schema = rfc2307bis
ldap_sasl_mech = GSSAPI
ldap_user_object_class = user
ldap_group_object_class = group
ldap_user_home_directory = unixHomeDirectory
ldap_user_principal = userPrincipalName
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true
ldap_user_search_base = cn=Users,dc=ntap,dc=local
ldap_group_search_base = cn=Users,dc=ntap,dc=local
ldap_sasl_username = CENTOS7$@NTAP.LOCAL
krb5_server = ntap.local
krb5_realm = NTAP.LOCAL
krb5_kpasswd = ntap.local
use_fullyQualified_names = false

[domain/NTAP.LOCALNTAP.LOCAL]
ad_domain = NTAP.LOCALNTAP.LOCAL
krb5_realm = NTAP.LOCAL
realm_tags = manages-system joined-with-adcli
cache_credentials = True
id_provider = ad
krb5_store_password_if_offline = True
default_shell = /bin/bash
ldap_id_mapping = True
use_fullyQualified_names = True
fallback_homedir = /home/%u@%d
access_provider = ad
```

The preceding configuration queries LDAP when no FQDN is provided, and when a user has an FQDN, the configuration uses AD. This approach delivers two different UIDs for the same user.

Without an FQDN, the following is returned:
With an FQDN, that same user gets the following:

```
# id prof1@NTAP.LOCAL
uid=1587401110(prof1@NTAP.LOCAL) gid=1587400513(domainusers@NTAP.LOCAL)
groups=1587400513(domainusers@NTAP.LOCAL),1587401105(group1@NTAP.LOCAL),1587401122(sharedgroup@NTAP.LOCAL),15874011106(group2@NTAP.LOCAL)
```

The numeric UID 1587401110 is generated based on the user's SID and the SSSD algorithm.

ONTAP currently does not support this method of UNIX user ID creation. The generated UID is not physically stored anywhere that ONTAP can query, and ONTAP does not currently use this algorithm to create UNIX user and group IDs.

When you use SSSD with ONTAP, either use LDAP as the ID provider and populate the LDAP server with the UNIX attributes, or create local passwd and group entries in the SVM with the same UID and GID information that the SSSD algorithm uses.

### DNS Considerations with SSSD

SSSD can use Kerberos authentication for secure LDAP binds. Therefore, you must configure DNS properly to include information about the LDAP universal resource identifier (URI) that is used in the SSSD configuration. SSSD does not support the use of round-robin DNS entries for failover. For failover to work properly, each entry must be unique and in DNS. For details, see the SSSD documentation.

Another limitation to SSSD is that failover depends on the order of entries in `/etc/resolv.conf`. If the first DNS server in the file is inaccessible, SSSD black-holes the attempt until the DNS server is available or until the `/etc/resolv.conf` file is modified. For more information about this point, see Red Hat bug 966757.

### 4.7 LDAP Authentication for Cluster Administration

You can also use LDAP servers to host users and groups that are used in cluster logins and authentication for CLI, API, and ONTAP System Manager access (users only).

This use is different from Active Directory domain tunneling. For the steps on how to do it, see the product documentation.

This section covers LDAP that uses FreeIPA for name services and KDC functionality on CentOS 8. These steps are not for setting up LDAP for user names and groups for file access. Instead, they are for logging in to the cluster for administration. However, many of the steps are identical to setting up LDAP for UNIX users and groups for file access.

### Basic Steps

At a high level, the following steps are how to set up LDAP for use with ONTAP for cluster logins. These steps are all performed on the cluster administration SVM, not on SVMs that are hosting data:

1. Verify that the password hashes that the LDAP server supports are among the supported methods in ONTAP:
   - CRYPT (all types), SHA, SSHA
2. Set up DNS on the cluster SVM.
3. Create the LDAP client schema from a template:
   - Be sure that the client schema matches your LDAP server.
4. Create an LDAP client that is owned by the cluster SVM:
   - Be sure that the client schema sets the bind user as one with permissions to view password hashes.
   - Be sure to set the bind password.
   - Be sure that the base, user, or group DN are set to the proper locations.
5. Enable LDAP on the cluster SVM.
6. Modify the ns-switch database to use LDAP.
7. Check LDAP lookups.
8. Create a security login account for the desired user or group with nsswitch as the authentication method.
9. Test logins.

**Detailed Steps**

The following section covers in further detail how to set up LDAP for user and group logins to ONTAP for cluster administration.

You can find the steps to create the FreeIPA server on CentOS 8 at the following link:

https://kifarunix.com/install-and-setup-freeipa-server-on-centos-8/

**FreeIPA Considerations**

With FreeIPA, there are a few things to consider when you try to use it for LDAP authentications through a password hash exchange.

**FreeIPA Schema—Compat**

FreeIPA, by default, creates two different CN areas that contain users. One is called compat, which is intended to provide backward compatibility for older LDAP and NIS environments. This CN can work fine for user and group lookups, but it does not contain userPassword fields, which are required for use with LDAP authentication in ONTAP.

For example, following is the output of ldapsearch from a user in CN=compat and the CN=accounts. Note that there is no userPassword attribute for the user in CN=compat.

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h 10.193.67.222 -b "dc=centos-ldap,dc=local" -s sub "(uid=idx-ldap)"
Enter LDAP Password:
# extended LDIF
# LDAPv3
# base <dc=centos-ldap,dc=local> with scope subtree
# filter: (uid=idx-ldap)
# requesting: ALL
#
# ldap://centos-ldap.corp.local
dn: uid=idx-ldap,cn=users,cn=compat,dc=centos-ldap,dc=local
objectClass: posixAccount
objectClass: ipaOverrideTarget
objectClass: top
gecos: IDM LDAP
cn: IDM LDAP
uidNumber: 1971600001
gidNumber: 1971600000
loginShell: /bin/sh
```
When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```

---

### When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```

---

### When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```

---

### When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```

---

### When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```

---

### When you set up the ONTAP LDAP client on the cluster SVM, you specify a base-dn field. Generally, this field is the top-level domain. However, with FreeIPA, it might cause the users and groups in the compat CN to be discovered first and then LDAP logins fail, even if you can properly query LDAP for user and group information.

To get around this issue, configure the user-dn and group-dn options in the LDAP client from the advanced privilege to use the proper user DNs that contain password information.

**LDAP client example for working authentication with FreeIPA:**

```bash
client::*> ldap client show -client-config IDM
```
Preferred Active Directory Servers: -
Bind Using the Vserver's CIFS Credentials: false
Schema Template: IPA
LDAP Server Port: 389
Query Timeout (sec): 3
Minimum Bind Authentication Level: simple

**Bind DN (User):** *uid=ONTAPLDAP,cn=sysaccounts,dc=centos-ldap,dc=local*
**Base DN:** *dc=centos-ldap,dc=local*
**Base Search Scope:** subtree
**User DN:** *cn=accounts,dc=centos-ldap,dc=local*
**User Search Scope:** subtree
**Group DN:** *cn=accounts,dc=centos-ldap,dc=local*
**Group Search Scope:** subtree
**Netgroup DN:** -
**Netgroup Search Scope:** subtree
**Vserver Owns Configuration:** false
**Use start-tls Over LDAP Connections:** false
**Enable Netgroup-By-Host Lookup:** false
**Netgroup-By-Host DN:** -
**Netgroup-By-Host Scope:** subtree
**Client Session Security:** none
**LDAP Referral Chasing:** false
**Group Membership Filter:** -

**FreeIPA—View Password Hashes in LDAP Queries**

You can modify FreeIPA LDAP schemas through an administrative user called the “Directory Manager.” This user can also see password hashes. By default, other users in the LDAP server cannot see password hashes, which is problematic for setting up LDAP authentication that requires comparison of password hashes, such as with ONTAP.

As a result, when you create an LDAP client, anonymous binds do not work, because anonymous binds do not have permissions to see password hashes. Also, when you set up a simple or SASL bind, it works only if you provide a user that has access to view password hashes, such as the Directory Manager. The FreeIPA best practices themselves state that you should not use the Directory Manager with remote services. The best practices linked page says to create a system account instead. The main goal here is the same: to have a bind user that can see the `userPassword` field in queries.

Following is an example of LDAP queries with users that can and cannot see the field.

**Nonworking bind example—no userPassword populated:**

```
cluster::*> ldap client show -client-config IDM -fields bind-dn
vserver client-config bind-dn
------- ------------------ ---------------
DEMO    IDM      idm-ldap

cluster::*> getxxbyyy getpwbyname -vserver cluster -username idm-ldap
(vserver services name-service getxxbyyy getpwbyname)
pw_name: idm-ldap
pw_passwd:
pw_uid: 1971600001
pw_gid: 1971600000
pw_gecos:
pw_dir:
pw_shell: /bin/sh
```

**Working bind example—userPassword populated:**

```
cluster::*> ldap client show -client-config IDM -fields bind-dn
vserver client-config bind-dn
------- ------------------ ---------------
DEMO    IDM      CN=Directory Manager

cluster::*> getxxbyyy getpwbyname -vserver cluster -username idm-ldap
(vserver services name-service getxxbyyy getpwbyname)
pw_name: idm-ldap
```

""
When you specify the newly created system account, be sure to use the full DN path. For instance:

```
-bd-dn uid=ONTAPLDAP, cn=sysaccounts, cn=etc, dc=centos, dc=local
```

If necessary, drop the `-min-bind-level` option to anonymous while you change the `-bind-dn` option. Or you can use the `-skip-configuration-validation true` option until you can modify the bind password or modify the bind password first.

**FreeIPA Schema—Password Hash Encryption Level**

When a user logs in with LDAP as the authentication method for ONTAP, a password hash comparison is performed as part of the login process. If the password hash encryption method is not supported by ONTAP, LDAP authentication fails. By default, FreeIPA uses the most secure password hash algorithm possible: SSHA512.

However, ONTAP currently supports only the following password hash lengths:

- CRYPT (all types)
- SHA
- SSHA

For SSH or web logins to work with LDAP authentication, when you create a user for authentication, the password hash must be one of the preceding lengths.

To change the password hash length for FreeIPA:

```
# ldapmodify -D "cn=Directory Manager" -w -p 389 -h centos8-IPA -x
dn: cn=config
changetype: modify
replace: passwordStorageScheme
passwordStorageScheme: CRYPT-SHA512
modifying entry "cn=config"
```

```
# ldapsearch -D "cn=Directory Manager" -w -p 389 -h centos8-IPA -b "cn=config" | grep passwordStorageScheme
Enter LDAP Password: 
passwordStorageScheme: CRYPT-SHA512
```

**Note:** CRYPT-SHA512 is currently the strongest supported password hash length for ONTAP.

You might not want to change the password hashes for all users, however. So, a workaround could be to modify the `passwordStorageScheme` value, create the desired users and passwords (or change existing passwords) with the new `passwordStorageScheme`, then set `passwordStorageScheme` back to the original value. Only new users or changed passwords would have the new `passwordStorageScheme` hash length.

**FreeIPA—LDAP Groups**

In some cases, you might want to use an LDAP group instead of a single user. The use of groups helps avoid extra overhead when you create login users on the cluster. By providing a centralized location, it also simplifies management of logins when you add or remove users. The one limitation of LDAP groups
for ONTAP cluster administration is that you currently can’t specify an LDAP group for use with HTTP or System Manager.

To get LDAP groups working with ONTAP logins, first and foremost, the group memberships must be properly seen from the ONTAP cluster. You can test it by using the `getxxbyyy` command set; see the section “GetXXbyYY.” Because FreeIPA uses the member LDAP scheme attribute for group memberships, you should modify the ONTAP LDAP client schema to reflect that aspect, as well as use RFC 2307bis.

The following schema template shows the custom schema to use with FreeIPA. It is based on the RFC 2307bis schema. Differences in schema attributes from the template are highlighted in yellow. Your schema template modifications might vary. For details, check with your LDAP administrator.

**Free IPA LDAP schema template example:**

```
Schema Template: IPA
  Comment:
  RFC 2307 posixAccount Object Class: person
  RFC 2307 posixGroup Object Class:posixgroup
  RFC 2307 nisNetgroup Object Class: nisNetgroup
    RFC 2307 uid Attribute:uid
    RFC 2307 uidNumber Attribute:uidNumber
    RFC 2307 gidNumber Attribute: gidNumber
  RFC 2307 cn (for Groups) Attribute:cn
  RFC 2307 cn (for Netgroups) Attribute:cn
  RFC 2307 userPassword Attribute:userPassword
  RFC 2307 gecos Attribute:gecos
  RFC 2307 homeDirectory Attribute:homeDirectory
  RFC 2307 loginShell Attribute:loginShell
  RFC 2307 memberUid Attribute: member
  RFC 2307 memberNisNetgroup Attribute:memberHost
  RFC 2307 nisNetgroupTriple Attribute: nisNetgroupTriple
    Enable Support for Draft RFC 2307bis: true
  RFC 2307bis groupOfUniqueNames Object Class:posixgroup
  RFC 2307bis uniqueMember Attribute: member

Data ONTAP Name Mapping windowsToUnix Object Class:posixAccount
  Data ONTAP Name Mapping windowsToUnix Object Class:posixAccount
    Data ONTAP Name Mapping windowsToUnix Account Attribute:windowsAccount
    Data ONTAP Name Mapping windowsToUnix Account Attribute:windowsAccount
    No Domain Prefix for windowsToUnix Name Mapping: false
      Vserver Owns Schema: false
    Maximum groups supported when RFC 2307bis enabled: 256
  RFC 2307 nisObject Object Class:ipahost
  RFC 2307 nisMapName Attribute: cn
  RFC 2307 nisMapEntry Attribute: cn
```

Following is an example of a working `getxxbyyy` command that queries group memberships:

```
cluster::*> getxxbyyy getgrlist -node node1 -vserver cluster -username ipa-user
  (vserver services name-service getxxbyyy getgrlist)
pw_name: ipa-user
Groups: 1971600000 1971600004

cluster::*> getxxbyyy getgrbygid -node node1 -vserver cluster -groupID 1971600000
  (vserver services name-service getxxbyyy getgrbygid)
name: admins
gid: 1971600000
gr_mem: admin uid=admin cn=users cn=accounts dc=centos-ldap dc=local

cluster::*> getxxbyyy getgrbygid -node node1 -vserver cluster -groupID 1971600004
  (vserver services name-service getxxbyyy getgrbygid)
name: ontap-ldap
gid: 1971600004
gr_mem: idm-ldap ipa-user uid=idm-ldap cn=users cn=accounts dc=centos-ldap dc=local uid=ipa-user cn=users cn=accounts dc=centos-ldap dc=local
```
After you have determined that LDAP group membership queries are populating the expected users, you can create security logins for the LDAP group. One difference from creating LDAP user accounts for logins is that you have to inform ONTAP that the accounts that you are creating are LDAP groups with the 
<code>-is-ns-switch-group yes</code> option.

```bash
cluster::*> security login create -user-or-group-name ontap-ldap -application ssh -authentication-method nsswitch -role admin -is-ns-switch-group yes -second-authentication-method none -vserver cluster
```

```bash
cluster::*> security login create -user-or-group-name ontap-ldap -application ontaip -authentication-method nsswitch -role admin -is-ns-switch-group yes -second-authentication-method none -vserver cluster
```

```bash
cluster::*> security login create -user-or-group-name ontap-ldap -fields is-ns-switch-group
vserver user-or-group-name application authentication-method is-ns-switch-group
------------------ ------------------ ------------------ ------------------ ------------------
cluster ontap-ldap ontapi nsswitch yes
cluster ontap-ldap ssh nsswitch yes
```

**LDAP Group Authentication Cache**

When an LDAP group is used for authentication, it gets cached in ONTAP. If you need to flush the cache, use the following:

```bash
cluster::*> security login group-authentication cache clear -vserver cluster -user ipa-user -application ssh
```

## 5 Common Issues and Troubleshooting Steps

The following sections cover common problems and troubleshooting steps to resolve LDAP issues in NetApp ONTAP. This section attempts to include as many scenarios and issues as possible, but it is not exhaustive. If you encounter an issue that is not covered in this section, be sure to collect the information as outlined in the section “What Information to Collect Before You Contact NetApp Support.” After you have all the pertinent information, contact NetApp technical support.

### 5.1 Optimize LDAP Searches

When you use ONTAP as an LDAP client for enterprise NAS, to eliminate delays in access, it is imperative that you make sure that the LDAP searches perform as quickly as possible. Although there is a copious amount of caching in ONTAP for NAS, there is still a cost associated with initial lookups. To enable optimal LDAP performance, you should follow the best practices that are outlined here. For a complete list of name service best practices, see [TR-4668: Name Services Best Practice Guide](#).

- Make sure that LDAP servers and associated name service servers (such as DNS) are on low-latency network links.
- Ideally, LDAP servers and DNS servers are local to the ONTAP cluster.
- Make sure that LDAP servers are not overworked (high CPU, and so on). Overworked LDAP servers return answers to queries more slowly. LDAP servers often have specific tools to measure performance, such as the [Active Directory Performance Testing Tool (ADTest.exe)](#). For more information about performance testing for LDAP, contact the LDAP server vendor.
- To troubleshoot search issues, use LDAP query tools such as `ldapsearch` or `ldp.exe`.
- To enable load balancing and redundancy, include multiple LDAP servers in any client configuration.
- Maintain your LDAP schemas to remove old records that are no longer in use.

Build LDAP schema structures and distinguished names (DNs) with a wide design rather than a deep design. Wide schemas allow shorter DNs. See Figure 33 for an example.
5.2 Points of Failure

When you attempt to get LDAP queries working with ONTAP, remember that an ONTAP SVM is acting as an LDAP client that connects to an LDAP server, just like any UNIX or Windows OS would. Therefore, when an issue occurs, there are multiple possible failure points for you to investigate.

**Network**

LDAP server connectivity starts and ends with the network. An ONTAP SVM provides IP addresses by way of data LIFs, virtual IP addresses that reside on physical network ports on a cluster node. Network failures can occur for various reasons, all of which are common network troubleshooting scenarios for any TCP/IP-dependent applications, for example:

- A network cable failed.
- A network port failed.
- The data LIF does not exist in the SVM.
- The network route is not configured for the SVM.
- Firewalls are blocking traffic.
- The data LIF resides on a network port that cannot reach the LDAP server’s network.
- The LDAP server’s network is down.

When the LDAP server is unreachable through a basic ping, ONTAP prevents LDAP configuration from being applied. For details on how to find network issues, see the “Troubleshooting Tools” section of this report.

**DNS**

DNS plays a part in LDAP failures in scenarios where LDAP host names are specified in the LDAP server list, where Active Directory domains are specified, or in cases where LDAP SRV records must be queried. DNS configuration is not required for use with LDAP, but NetApp highly recommends it. DNS failures can occur for multiple reasons, including, but not limited to:

- Network connectivity issues
- DNS query timeouts
- Hosts that are not found
• Incorrect DNS configuration

When the DNS server is unreachable through a basic ping, ONTAP prevents DNS configuration from being applied. For details on how to find DNS issues, see the “Troubleshooting Tools” section.

Name Service Switch (ns-switch) Configuration

Name service switches tell clients where they should look for various name services and in what order they should look. ONTAP SVMs each have their own ns-switch configurations to specify external name services or local files for user, group, netgroup, and host name lookups.

To make ONTAP SVMs able to communicate with LDAP servers, ns-switch configuration is a required step; see the section “Modify the SVM Name Service Switch (ns-switch).”

LDAP Ports

LDAP port configuration is how ONTAP SVMs know which network ports to use for communication with LDAP servers. Port 389 is the standard LDAP port, but in some environments, the LDAP port might be different based on LDAP server configuration. For instance, the use of LDAP over SSL requires port 636 in many cases. In other cases, LDAP ports might be changed to nonstandard LDAP ports for security considerations through obfuscation. ONTAP SVMs support only port 636 if you use LDAP over SSL, and they support only port 389 if you use LDAP with StartTLS. Standard LDAP calls can use any port as configured on the server and client. To verify which port is being used for LDAP communication, check with the LDAP administrator.

LDAP Binds

An LDAP bind is the way that LDAP clients “log in” to an LDAP server to perform read-only queries for name and group lookups. In most cases, any user in LDAP can bind to the LDAP server to read the schema attributes. In the case of user password fields, a privileged LDAP user might be required (such as the Directory Manager in Linux-based LDAP for cluster administration by using LDAP logins).

Binds can be set up in three different ways, but LDAP servers might be configured to support only specific ways of binding:

• Anonymous, which allows anyone to read the LDAP schema
• Simple, which is the basic user name and password bind
• SASL, which is the most secure form of binding and can be done through various encryption methods, including Kerberos and NTLM

If an LDAP bind fails, then user and group lookups also fail. If an LDAP bind fails, ONTAP logs messages to the event log, as does the LDAP server. For details on how to find LDAP bind issues, see the “Troubleshooting Tools” section.

LDAP DN Search Configuration

DN search configuration refers to the schema locations that you specify in LDAP client configurations to help direct LDAP searches to look in the most efficient places. The base DN is specified to direct the main searches, but user, group, and netgroup DN filters also help narrow down queries further to help reduce the amount of time that is spent on looking up objects.

If an LDAP DN is misconfigured or if the wrong DN is specified, then searches for users and groups succeed but do not return the expected results (or any results at all). An analogy is looking for someone named “Smith” in the “R” section of a phone book.

Issues with LDAP DN configuration can be difficult to track down. There are no errors other than object not found, so troubleshooting requires a bit more digging. For details on how to find LDAP DN issues, see the “Troubleshooting Tools” section.
LDAP Client Schema Configuration

ONTAP provides several default, read-only LDAP Schema Templates for use with LDAP client configuration. For ease of use, these templates offer common schema attribute values. In some cases, however, an LDAP server might be using a schema configuration that does not match the templates. In those cases, you can copy schema templates to new writeable templates for modification.

These schema attributes help formulate the LDAP search queries that ONTAP generates for user and group lookups. If the wrong schema attributes are specified, then ONTAP sends the wrong values and does not retrieve the proper results from the LDAP server. This result causes file access issues, because ONTAP cannot discern who a user is based on the incoming client request and cannot apply the proper permissions to the user.

Incorrect schema configurations can also result in incorrect secondary group memberships. To help get the schema attributes right, be sure to follow the instructions in the “LDAP Schema Configuration” section of this report.

For details on how to discover LDAP schema configuration issues, see the “Troubleshooting Tools” section.

5.3 Troubleshooting Tools

This section covers specific tools and commands that you can use to troubleshoot LDAP issues in your environment.

Third-Party Tools and Utilities

In some cases, troubleshooting LDAP issues require third-party utilities. This section covers a few of those tools.

Basic Network Troubleshooting: ping, nslookup, dig, telnet, nmap

Basic troubleshooting for LDAP generally calls for checking network connectivity. Pings can check whether networks are connecting properly between the LDAP server, client, and ONTAP. Basic pings, however, might sometimes be blocked in networks, so a utility such as telnet or nmap to check port connectivity can be useful. Also, checking name resolution with DNS is important in some LDAP configurations, so you can use a tool such as nslookup or dig to test DNS.

Example of nmap to an LDAP server that is running Windows 2012 R2:

```
# nmap x.x.x.x

Starting Nmap 6.40 ( http://nmap.org ) at 2020-02-14 23:34 EST
Nmap scan report for oneway.ntap.local (x.x.x.x)
Host is up (0.00023s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
53/tcp    open  domain
80/tcp    open  http
88/tcp    open  kerberos-sec
111/tcp   open  rpcbind
135/tcp   open  mprpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
2049/tcp  open  nfs
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
3389/tcp  open  ms-wbt-server
```
Example of `nslookup` and `dig`:

```bash
# nslookup ntap.local
Server:        x.x.x.x
Address:       x.x.x.x#53

Non-authoritative answer:
Name:  ntap.local
Address: x.x.x.x

# dig ntap.local
; <<>> DiG 9.9.4-RedHat-9.9.4-61.el7_5.1 <<>> ntap.local
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9482
; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
; QUESTION SECTION:
;ntap.local.

; ANSWER SECTION:
ntap.local.             590     IN      A       x.x.x.x
; Query time: 1 msec
; SERVER: x.x.x.x#53(x.x.x.x)
; WHEN: Fri Feb 14 23:37:57 EST 2020
; MSG SIZE  rcvd: 5

Use of `dig` to query SRV records:

```bash
# dig SRV ldap/oneway.ntap.local
; <<>> DiG 9.9.4-RedHat-9.9.4-61.el7_5.1 <<>> SRV ldap/oneway.ntap.local
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 31583
; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
; QUESTION SECTION:
;ldap/oneway.ntap.local.

; AUTHORITY SECTION:
ntap.local.             900     IN      SOA      oneway.ntap.local. hostmaster.ntap.local. 2399900 600 86400 3600
; Query time: 1 msec
; SERVER: x.x.x.x#53(x.x.x.x)
; WHEN: Fri Feb 14 23:39:09 EST 2020
; MSG SIZE  rcvd: 105

# dig SRV _ldap._tcp.ntap.local
; <<>> DiG 9.9.4-RedHat-9.9.4-61.el7_5.1 <<>> SRV _ldap._tcp.ntap.local
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62440
; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2

; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
; QUESTION SECTION:
;_ldap._tcp.ntap.local.

; ANSWER SECTION:
ldap._tcp.ntap.local. 600 IN  SRV 0 100 389 oneway.ntap.local.
```

Packet Traces

In many cases, packet traces can be invaluable for isolating a wide array of LDAP issues—from basic connectivity, to bind issues, to LDAP search issues. You can use any packet tracing utility from tcpdump to Wireshark. In cases where LDAP traffic is encrypted through SSL or certificates, packet traces are not as useful, but you can sometimes use the cert keys to decrypt traces. The following links give some examples of this approach:

- IDMWORKS: Decrypting LDAPS traffic to Active Directory
- Microsoft Docs: Reading LDAP SSL Network Traffic with NetMon 3.4 and NMDecryp
- Wireshark: Trouble decoding LDAP over SSL

For an example of LDAP communication as seen through packet capture, see the section “LDAP Traffic as Seen from a Packet Trace.”

LDAP Browser—Softerra

Softerra offers a free LDAP browser application that allows you to see LDAP schemas, test binds and queries, and so on. This browser can use all the same types of binds that ONTAP uses, including SSL certificates, and is useful for finding which attributes you need for custom LDAP client schemas. See Figure 34.

Figure 34) Softerra LDAP browser.

Ldp

Ldp is an LDAP client the Microsoft Windows provides. This utility has functionality that is similar to the Softerra LDAP browser, but without as much GUI interaction. This tool is usually found on Active Directory domain controllers, but it can also be installed on Windows clients.

Ldp enables you to:
Connect
Bind
Use StartTLS and/or LDAP over SSL
Browse
Run search queries

This functionality provides a way for you to test LDAP functionality outside of ONTAP so that you can confirm that the LDAP servers are working properly and are honoring search queries. See Figure 35 for a sample search.

Figure 35) Sample LDAP search by using Ldp.

Figure 35

ldapsearch

ldapsearch is a standard Linux-based utility that allows command-line interaction with LDAP servers to perform standard LDAP searches. This command allows you to specify bind level, certificates to use, search filters, and various other useful mechanisms that assist in LDAP server and schema troubleshooting.

Example of ldapsearch:

```
# ldapsearch -h x.x.x.x -p 389 -x -b 'dc=centos-ldap,dc=local' -s sub '(uid=ipa-user)'
# extended LDIF
#
# LDAPv3
# base <dc=centos-ldap,dc=local> with scope subtree
# filter: (uid=ipa-user)
# requesting: ALL
#
# ipa-user, users, accounts, centos-ldap.local
dn: uid=ipa-user,cn=users,cn=accounts,dc=centos-ldap,dc=local
givenName: IPA
sn: User
uid: ipa-user
cn: IPA User
displayName: IPA User
initials: IU
gecos: IPA User
gidNumber: 1971600000
objectClass: top
objectClass: person
```
objectClass: organizationalperson
objectClass: inetOrgPerson
objectClass: inetuser
objectClass: posixAccount
objectClass: krbprincipalaux
objectClass: krbticketpolicyaux
objectClass: ipaObject
objectClass: ipasshuser
objectClass: ipauser
objectClass: ipasshGroupOfPubKeys
objectClass: mepOriginEntry
loginShell: /bin/sh
homeDirectory: /home/ipa-user
uidNumber: 1971600003

**PowerShell**

If you use Active Directory for your UNIX LDAP environment, you can use PowerShell to dump attributes for users and groups. A list of attributes of users and groups is invaluable for creating and troubleshooting LDAP client schema issues.

**Example of UNIX user and group attributes dump by using PowerShell:**

```powershell
PS C:\> Get-ADUser prof1 -properties *

AccountExpirationDate : 9223372036854775807
accountExpires : 9223372036854775807
AccountLockoutTime : False
AccountNotDelegated : False
AllowReversiblePasswordEncryption : False
AuthenticationPolicy : {}
AuthenticationPolicySilo : {} 
BadLogonCount : 0
badPasswordTime : 132146010015585937
badPwdCount : 0
CannotChangePassword : False
CanonicalName : NTAP.LOCAL/Users/prof1
Certificates : {}
City : 
CN : prof1
codePage : 0
Company : 
CompoundIdentitySupported : {} 
Country : 
countryCode : 0
Created : 1/14/2017 12:23:19 PM
createTimestamp : 1/14/2017 12:23:19 PM
Deleted : 
Department : 
Description : 
DisplayName : prof1
DistinguishedName : CN=prof1,CN=Users,DC=NTAP,DC=local
Division : 
DoesNotRequirePreAuth : False
dSCorePropagationData : 12/31/1600 7:00:00 PM
EmailAddress : 
EmployeeID : 
EmployeeNumber : 
Enabled : True
Fax : 
gecos : Prof1
gidNumber : 1101
GivenName : prof1
HomeDirectory : 
HomeDirRequired : False
HomeDrive : 
HomePage : 
HomePhone : 
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanonicalName</td>
<td>NTAP.LOCAL/Users/ProfGroup</td>
</tr>
<tr>
<td>CN</td>
<td>ProfGroup</td>
</tr>
<tr>
<td>Created</td>
<td>1/14/2017 12:24:31 PM</td>
</tr>
<tr>
<td>createTimeStamp</td>
<td>1/14/2017 12:24:31 PM</td>
</tr>
<tr>
<td>Deleted</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>DisplayName</td>
<td></td>
</tr>
<tr>
<td>DistinguishedName</td>
<td>CN=ProfGroup,CN=Users,DC=NTAP,DC=local</td>
</tr>
<tr>
<td>dSCorePropagationData</td>
<td>(12/31/1600 7:00:00 PM)</td>
</tr>
<tr>
<td>gidNumber</td>
<td>1101</td>
</tr>
<tr>
<td>GroupCategory</td>
<td>Security</td>
</tr>
<tr>
<td>GroupScope</td>
<td>Global</td>
</tr>
<tr>
<td>groupType</td>
<td>-2147483646</td>
</tr>
<tr>
<td>HomePage</td>
<td></td>
</tr>
<tr>
<td>instanceType</td>
<td>4</td>
</tr>
<tr>
<td>isDeleted</td>
<td></td>
</tr>
<tr>
<td>LastKnownParent</td>
<td></td>
</tr>
<tr>
<td>ManagedBy</td>
<td>(CN=quota user,CN=Users,DC=NTAP,DC=local, CN=prof1,CN=Users,DC=NTAP,DC=local, CN=student2,CN=Users,DC=NTAP,DC=local, CN=Administrator,CN=Users,DC=NTAP,DC=local)</td>
</tr>
<tr>
<td>MemberOf</td>
<td>{}</td>
</tr>
<tr>
<td>Members</td>
<td>(CN=quota user,CN=Users,DC=NTAP,DC=local, CN=prof1,CN=Users,DC=NTAP,DC=local, CN=student2,CN=Users,DC=NTAP,DC=local, CN=Administrator,CN=Users,DC=NTAP,DC=local)</td>
</tr>
<tr>
<td>Modified</td>
<td>8/30/2019 3:38:12 PM</td>
</tr>
<tr>
<td>modifyTimeStamp</td>
<td>8/30/2019 3:38:12 PM</td>
</tr>
<tr>
<td>Name</td>
<td>ProfGroup</td>
</tr>
<tr>
<td>ObjectCategory</td>
<td>CN=Group,CN=Schema,CN=Configuration,DC=NTAP,DC=local</td>
</tr>
<tr>
<td>ObjectClass</td>
<td>group</td>
</tr>
<tr>
<td>ObjectGUID</td>
<td>e2fae6f6-e682-4c37-b998-d0e2215e8e66</td>
</tr>
<tr>
<td>objectSid</td>
<td>5=1-5-21-3552729481-4032800560-2279794651-111</td>
</tr>
<tr>
<td>ProtectedFromAccidentalDeletion</td>
<td>False</td>
</tr>
<tr>
<td>SamAccountName</td>
<td>ProfGroup</td>
</tr>
<tr>
<td>sAMAccountType</td>
<td>268435456</td>
</tr>
<tr>
<td>sDRightsEffective</td>
<td>15</td>
</tr>
<tr>
<td>SID</td>
<td>5=1-5-21-3552729481-4032800560-2279794651-111</td>
</tr>
<tr>
<td>SIDHistory</td>
<td>{}</td>
</tr>
<tr>
<td>uSNChanged</td>
<td>680730</td>
</tr>
<tr>
<td>uSNCreated</td>
<td>12933</td>
</tr>
<tr>
<td>whenChanged</td>
<td>8/30/2019 3:38:12 PM</td>
</tr>
<tr>
<td>whenCreated</td>
<td>1/14/2017 12:24:31 PM</td>
</tr>
</tbody>
</table>

For a search of netgroup attributes, use `Get-AdObject`:

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanonicalName</td>
<td>NTAP.LOCAL/netgroups/netgroup1</td>
</tr>
<tr>
<td>CN</td>
<td>netgroup1</td>
</tr>
<tr>
<td>Created</td>
<td>3/1/2017 3:04:00 PM</td>
</tr>
<tr>
<td>createTimeStamp</td>
<td>3/1/2017 3:04:00 PM</td>
</tr>
<tr>
<td>Deleted</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>DisplayName</td>
<td></td>
</tr>
<tr>
<td>DistinguishedName</td>
<td>CN=netgroup1,OU=netgroups,DC=NTAP,DC=local</td>
</tr>
<tr>
<td>dSCorePropagationData</td>
<td>(12/31/1600 7:00:00 PM)</td>
</tr>
<tr>
<td>instanceType</td>
<td>4</td>
</tr>
<tr>
<td>isDeleted</td>
<td></td>
</tr>
<tr>
<td>LastKnownParent</td>
<td></td>
</tr>
<tr>
<td>Modified</td>
<td>2/20/2020 9:43:17 PM</td>
</tr>
<tr>
<td>modifyTimeStamp</td>
<td>2/20/2020 9:43:17 PM</td>
</tr>
<tr>
<td>Name</td>
<td>netgroup1</td>
</tr>
<tr>
<td>nisNetgroupTriple</td>
<td>((10.193.67.225,,), (xcp,))</td>
</tr>
<tr>
<td>ObjectCategory</td>
<td>CN=nisNetgroup,CN=Schema,CN=Configuration,DC=NTAP,DC=local</td>
</tr>
<tr>
<td>ObjectClass</td>
<td>nisNetgroup</td>
</tr>
</tbody>
</table>
ONTAP CLI Commands for LDAP Troubleshooting

ONTAP provides several commands and log files that you can use to troubleshoot LDAP issues. This section covers those commands.

Network Pings

ONTAP can run network pings to any destination host name or address, including a way to force the traffic out of a specific data LIF and SVM.

```
cluster::*> network ping ?
{ -node <nodename>                   Node
  -lif <lif-name>                   Logical Interface
  -use-source-port {true|false}     *(DEPRECATED)-Use Source Port of Logical Interface
  [-destination] <Remote InetAddress> Destination
  [-show-detail]-s [true]          Show Detail Output
  [-record-route|-R [true]         Record Route
  [-verbose|-v [true]              Show All ICMP Packets
  [-packet-size <integer>]         Packet Size
  [-count <integer>]               Count
  [-wait <integer>]                Packet Send Wait Time (secs)
  [-flood [true]                   *Flood Ping
  [-disallow-fragmentation|-D [true] Disallow Packet Fragmentation (default: false)
```

GetXXbyYY

ONTAP provides a command in the advanced privilege (getxxbyyy) that allows searches to be performed from an SVM to the name services that are configured in the ns-switch file for the SVM.

Following are the available commands:

```
cluster::*> getxxbyyy ?
  (vserver services name-service getxxbyyy)
getaddrinfo           *Gets the IP address information by using the host name.
getgrbygid            *Gets the group members by using the group identifier or GID.
getgrbyname           *Gets the group members by using the group name.
getgrlist             *Gets the group list by using the user name.
gethostbyaddr         *Gets the host information from the IP address.
gethostbyname         *Gets the IP address information from host name.
getnameinfo           *Gets the name information by using the IP address.
getpwnbyname          *Gets the password entry by using the user name.
getpwnbyname          *Gets the password entry by using the user identifier or UID.
netgrpcheck           *Check if a client is part of a netgroup using combined API
```

getxxbyyy can query LDAP, NIS, or local files for users, groups, group memberships, and netgroups and can query DNS or local files for host names. In addition, the command can specify which name service source the results came from (-show-source true) and can provide granular errors from the commands (hidden option -show-granular-err true). You can also bypass the cache to confirm that lookups are coming from name services with -use-cache false.

Example of getxxbyyy to look up a UNIX user:

```
cluster::*> getxxbyyy getpwnbyname -vserver NFS -node node1 -username ipa-user -show-source true -show-granular-err true
```
(vserver services name-service getxxbyyyy getpwbyname)
Source used for lookup: LDAP
pw_name: ipa-user
pw_passwd:
| (crypt)$6$GSKkUw19eRKmzY8RQwVywzNW0as5xFPV121B5PZa3.soXlnQFmmTCuojtZ/H9dqt6vjUBHS4V2fKZ93Z9
| pw_uid: 1971600003
| pw_gid: 1971600000
| pw_gecos: /home/ipa-user
| pw_dir: /bin/sh

NIS:
Error code: NS_ERROR_NONE
Error message: No error
LDAP:
Error code: NS_ERROR_NONE
Error message: No error
DNS:
Error code: NS_ERROR_NONE
Error message: No error
FILES:
Error code: NS_ERROR_NONE
Error message: No error
Determined Result: Success

Example of getxxbyyyy to retrieve group membership:
cluster::*-> getxxbyyyy getgrlist -node node1 -vserver DEMO -username seventeengids
pv_name: seventeengids
Groups: 1201 12348 123411 123415 12345 12344 123414 12349 123412 123417 1234 123411 123410 123413 12343 123412 12342 1202 1203 1204 1220 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216 1217

Example of host name lookup with getxxbyyyy:
cluster::*-> getxxbyyyy gethostbyname -node node1 -vserver DEMO -hostname centos8 -show-source true
Source used for lookup: DNS
Host name: centos8
Canonical name: centos8.NTAP.LOCAL
IPv4: x.x.x.x

Example of netgroup member check:
cluster::*-> getxxbyyyy netgrpcheck -node node1 -vserver DEMO -netgroup netgroup1 -clientIP 10.193.67.225 -show-source true
Success. Client 10.193.67.225 is member of netgroup netgroup1
Searched using NETGROUP_BYNAME
Source used for lookup: LDAP

vserver services access-check
ONTAP 9.3 and later versions offer commands for interaction with name services, including:

- Authentication
- DNS
- Name mapping
- Server discovery

These commands live under the vserver services access-check command set. The access-check commands are essentially diag secd commands that are ported to the advanced privilege.

Authentication
Authentication commands allow storage administrators to review users, groups, group memberships, numeric IDs, and name mappings for multiprotocol NAS environments. These commands are useful mostly to verify that users and IDs match their expected results and to check whether group membership information is working across both SMB and NFS protocols.

One of the more detailed commands is the `show-creds` command. This command works only if NFS and SMB are configured for the SVM, but it offers a wide array of information about the user that is being queried, including numeric IDs, SIDs, name mappings, and so on.

**Example of `show-creds` command:**

```bash
cluster::*> vserver services access-check authentication show-creds -vserver DEMO -win-name prof1 -list-id true -list-name true

UNIX UID: 1100 (prof1) <> Windows User: S-1-5-21-3552729481-4032800560-22779794651-1110 NTAP\prof1 (Windows Domain User) << Name Mapping

GID: 1101 (ProfGroup) << Primary UNIX group
Supplementary GIDs: << UNIX group membership
  1101 (group1)
  1201 (group2)
  1203 (group3)
  1220 (sharedgroup)

Primary Group SID: S-1-5-21-3552729481-4032800560-22779794651-513 NTAP\DomainUsers (Windows Domain group)

Windows Membership: << Windows SMB group membership
S-1-5-21-3552729481-4032800560-22779794651-1106 NTAP\group2 (Windows Domain group)
S-1-5-21-3552729481-4032800560-22779794651-513 NTAP\DomainUsers (Windows Domain group)
S-1-5-21-3552729481-4032800560-22779794651-1112 NTAP\sharedgroup (Windows Domain group)
S-1-5-21-3552729481-4032800560-22779794651-1105 NTAP\group1 (Windows Domain group)
S-1-5-21-3552729481-4032800560-22779794651-1107 NTAP\group3 (Windows Domain group)
S-1-5-21-3552729481-4032800560-22779794651-1111 NTAP\ProfGroup (Windows Domain group)
S-1-18-2 Service asserted identity (Windows Well known group)
S-1-5-32-551 BUILTIN\Backup Operators (Windows Alias)
S-1-5-32-545 BUILTIN\Users (Windows Alias)
User is also a member of Everyone, Authenticated Users, and Network Users

Privileges (0x2086): << Windows privileges
SeBackupPrivilege
SeRestorePrivilege
SeChangeNotifyPrivilege
```

**DNS**

The `access-check` command can also query DNS for forward or SRV lookups. Alternatively, you can use the `getxxbyyy` command.

**Name Mapping**

Name Mapping in ONTAP occurs for users in multiprotocol NAS environments. The intent is to authenticate users based on the Security Styles of the volumes or the qtree that is being accessed, because a valid user must be present to determine access permissions based on the ACL styles on the file or folder. (See the “Security Styles” section.) For instance, with NTFS security styles, UNIX users must map to valid Windows users. LDAP helps centralize users and groups to provide more seamless name mapping for multiprotocol NAS environments.

The `access-check name-mapping` command allows storage administrators to verify that name mappings are correct for UNIX and Windows users, as well as for Kerberos SPN mappings for Kerberized NFS access.

**Example of `access-check name-mapping` command:**
vserver services name-service

Another set of commands that can help with LDAP and NAS troubleshooting are the vserver services name-service commands. The following set of commands are available in the advanced privilege:

```
cluster::*> vserver services name-service

cache> *The cache directory
dns> Manage DNS service
getxxbyyy> *Execute getXXbyYY for the given command.
ldap> Manage LDAP configuration
netgroup> Manage local netgroups
nis-domain> Manage Network Information Service domains
ns-switch> Manage Name Services Switch ordering
unix-group> Manage local UNIX group accounts
unix-user> Manage local UNIX user accounts
ypbind> *The ypbind directory
```

This section discusses these commands and where they fit into LDAP and NAS troubleshooting.

cache

To help reduce the overall load on networks and name servers, ONTAP caches many name service requests. These caches are covered in detail in TR-4668. With these commands, you can check, flush, and configure cache lifetimes.

Following are the available caches that you can manage:

```
cluster::*> name-service cache

- group-membership hosts
- netgroups
- settings
- show-count
- unix-group
- unix-user
```

**Example of group-membership cache:**

```
cluster::*> name-service cache group-membership show -vserver DEMO -user prof1

<table>
<thead>
<tr>
<th>Vserver</th>
<th>User</th>
<th>Group</th>
<th>Groups</th>
<th>Create Time</th>
<th>Is Partial</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>prof1</td>
<td>1101</td>
<td>5</td>
<td>1101, 1201, 1202, 1203, 1220</td>
<td>False</td>
</tr>
</tbody>
</table>

You can also delete a group membership cache in the SVM on a per-user basis. This capability comes in handy if a user has been added or removed from a group recently and the cache does not accurately reflect that change.

Cache views and flushes also can be applied to DNS host names, UNIX users and groups, and netgroups.

**Example of DNS host name cache:**

```
cluster::*> name-service cache hosts forward-lookup show -vserver DEMO -host *

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Host</th>
<th>Protocol</th>
<th>Family</th>
<th>Address</th>
<th>Source</th>
<th>Time</th>
<th>TTL(sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>oneway.ntap.local</td>
<td>Any</td>
<td>Any</td>
<td>xx.xxx.xx.xxx</td>
<td>dns</td>
<td>2/13/2020</td>
<td>3600</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15:39:27</td>
<td></td>
</tr>
</tbody>
</table>
```

* NTAP\prof1' maps to 'prof1'
Example of unix-user cache:

```
cluster::*> name-service cache unix-user user-by-name show -vserver DEMO -pw-name prof1
(vserver services name-service cache unix-user user-by-name show)

  Vserver: DEMO
  pw_name field: prof1
  pw_uid field: 1100
  pw_gid field: 1101
  Create Time: 2/13/2020 15:27:30
Source of the Entry: ldap
```

When a netgroup query is run for a single host, ONTAP populates the cache with all the hosts from the netgroup, which reduces the amount of load on the cluster. When you check the caches, you can see the ip-to-netgroup cache of specific hosts that were queried:

```
cluster::*> name-service cache netgroups ip-to-netgroup show -vserver DEMO
(vserver services name-service cache netgroups ip-to-netgroup show)

<table>
<thead>
<tr>
<th>Vserver</th>
<th>IP Address</th>
<th>Netgroup</th>
<th>Source</th>
<th>Create Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>10.193.67.222</td>
<td>netgroup1</td>
<td>ldap</td>
<td>2/20/2020 21:21:21</td>
</tr>
<tr>
<td>DEMO</td>
<td>10.193.67.233</td>
<td>netgroup1</td>
<td>netgrp_byname</td>
<td>2/20/2020 21:14:12</td>
</tr>
</tbody>
</table>
```

Or you can see the full list of members, whether they were queried or not:

```
cluster::*> name-service cache netgroups members show -vserver DEMO
(vserver services name-service cache netgroups members show)

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Netgroup</th>
<th>Hosts</th>
<th>Source</th>
<th>Create Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>netgroup1</td>
<td>10.193.67.225,xcp.ntap.local,xcp</td>
<td>ldap</td>
<td>2/20/2020 21:08:58</td>
</tr>
</tbody>
</table>
```

Caches all have a specific timeout period (Time to Live, or TTL), after which entries age out to prevent stale entries from lingering. There are also negative TTL values, where a lookup that has failed resides to prevent overrunning a system with objects that might not exist. You can adjust the cache timeout values by using the settings command under each cache type. You can also enable or disable caches.

Table 13 shows the default cache timeout values for each cache in ONTAP 9.7.

Table 13) Default cache timeout values in ONTAP 9.7.

<table>
<thead>
<tr>
<th>Cache</th>
<th>Default Timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group membership list</td>
<td>24-hour TTL</td>
</tr>
<tr>
<td>Hosts</td>
<td>24-hour TTL, 1-minute negative TTL</td>
</tr>
<tr>
<td>Netgroups</td>
<td>24-hour TTL, 30-minute negative TTL</td>
</tr>
<tr>
<td>UNIX groups</td>
<td>24-hour TTL, 1-minute negative TTL</td>
</tr>
<tr>
<td>UNIX users</td>
<td>24-hour TTL, 1-minute negative TTL</td>
</tr>
</tbody>
</table>

Note: Cache timeout values might change depending on the ONTAP release. Be sure to verify the cache settings for your ONTAP release.
In addition, the name service cache also undergoes a total cache eviction every 4 hours by default, but you can control this period with the global name-service cache settings command. This command also can control whether or not the name-service cache replicates between nodes to provide a global name-service cache for SVMs.

**dns**

With name-service dns commands, you can configure DNS or local host file entries, enable dynamic DNS updates, or check the DNS server status.

For purposes of troubleshooting, dns check can provide information about connectivity and the speed of a request. When you run this command, a standard “A” record query for example.dnsdomain.com is performed to the configured DNS server, as shown in Figure 36.

**Figure 36) Configured DNS server.**

![Configured DNS server](image)

**Example of dns check:**

```
cluster::*> name-service dns check -vserver DEMO
 (vserver services name-service dns check)

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Name Server</th>
<th>Status</th>
<th>Status Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>xx.xxx.xx.xxx</td>
<td>up</td>
<td>Response time (msec): 1</td>
</tr>
</tbody>
</table>
```

**Note:** By default, dns check is run when DNS configurations are created. If dns check fails during any of the tests, then the configuration modification fails, unless -skip-config-validation is set to true.

**getxxbyyy**

This command is covered in the previous section “GetXXbyYY” and is used for name-service checks for users, groups, and DNS lookups.

**ldap**

With name-service ldap commands, you can configure LDAP clients and schemas or check the LDAP server status.

For purposes of troubleshooting, ldap check can provide information about connectivity and the speed of a request. When you run this command, the following occurs:

- Basic connectivity to the LDAP server is checked:
  - If a host name is set in -ldap-servers, DNS is queried to resolve to an IP address.
  - If an IP address is configured, DNS is not used.
  - If the -ad-domain setting is configured, then DNS searches for the LDAP SRV record.
- An LDAP bind takes place based on the LDAP client configuration:
  - The minimum bind level is the minimum bind level that is allowed. More secure bind methods are tried first, starting with SASL.
- If the bind is successful, the configured DNs are checked:
  - Queries are performed if base, user, group, and/or netgroup DNs are configured.
Queries are basic wholeSubtree and are looking only for the LDAP server to respond that the DNs exist.

Note: By default, ldap check is run when LDAP client configurations are created. If ldap check fails during any of the tests, then the configuration modification fails, unless –skip-config-validation is set to true.

Example of ldap check:

```
cluster::> name-service ldap check -vserver DEMO
(vserver services name-service ldap check)

Vserver: DEMO
Client Configuration Name: DEMO
  LDAP Status: up
  LDAP Status Details: Successfully connected to LDAP server "x.x.x.x".
  LDAP DN Status Details: All the configured DNs are available.
```

netgroup

The netgroup commands allow netgroup files to be loaded through the URI from external sources to caches on the local SVM for faster access. Files are loaded through FTP or HTTP.

Example of netgroup load:

```
cluster::*> netgroup load -vserver NFS -source http://x.x.x.x/files/netgroupfile.txt
[Job 25720] Job succeeded: Netgroup Load Job Success
cluster::*> netgroup file show -vserver NFS

Vserver: DEMO
Netgroup File: netgroupfile

```

nis-domain

This command set is unrelated to LDAP and instead is used for interaction with NIS.

ns-switch

This command set is used to manage the ns-switch configuration for an SVM, such as specifying LDAP as a source for user and group lookups.

unix-user and unix-group

These command sets are used to manage local users and groups, as well as to enable file-only mode for users and groups. For details on file-only mode, see the section “Use of Local Files in ONTAP.”

ypbind

This command is a standard NIS command that is used to test connectivity to NIS servers. It does not apply to LDAP.

export-policy

Export policies in ONTAP are containers for export rules for NFS (or, if desired, SMB) share access. Unlike share or file permissions, access is controlled by client IP address, host name, netgroup, or subnet. Export policy rules are intended to be an initial gating option and should not be considered as the sole way to secure NFS. You should also use other methods, such as Kerberos, encryption, and user permissions.
This section covers commands under the `export-policy` set that pertain to LDAP functionality, because LDAP can serve netgroups to the cluster, which can then be applied to exports.

**check-access**

The `check-access` command enables you to specify volumes, qtrees, and client IP addresses to check the level of access that is allowed to the export. ONTAP resolves DNS host names, searches LDAP for netgroups, and sends test operations to the export during this process.

The following options are available with the command:

```bash
cluster:~$ export-policy check-access ?
        [-instance | -fields <fieldname>, ... ]
        -vserver <vserver name> Vserver Name
        [-volume] <volume name> Volume Name
        [-client-ip] <IP Address> Client IP Address
        [-authentication-method] <authentication method> Authentication Method
        [-access-type] {read|read-write} Access Rights to Check for
          [-qtree <qtree name>] Name of the Qtree
          [-path <text>] Path
          [-policy <text>] Export Policy
          [-policy-owner <text>] Export Policy Owner
          [-policy-owner-type {volume|qtree}] Type of Export Policy Owner
          [-rule-index <integer>] Export Policy Rule Index
          [-access {read|read-write}] Access Rights
          [-partial-rule-match [true|false]] Did a Subset of the Rules Match?
          [-clientmatch <text>] Client Match Spec
```

This example tests an export policy that has a netgroup in the rule set.

```bash
cluster:~$ vol show -vserver DEMO -volume netgrpvol -fields policy
vserver volume policy
------------------------
DEMO netgrpvol netgroup
cluster:~$ export-policy rule show -vserver DEMO -policyname netgroup
Policy Rule Access Client RO
Vserver Name Index Protocol Match Rule
------------------------
DEMO netgroup 1 any @netgroup1 any
cluster:~$ export-policy check-access -vserver DEMO -volume netgrpvol -client-ip 10.193.67.222 -authentication-method sys -protocol nfs3 -access-type read-write
Path Policy Policy Rule
------------------------
/netgrpvol default vsroot volume 2 read
```

You can see that the netgroup cache is populated after the command is run:

```bash
cluster:~$ cache netgroups ip-to-netgroup show -vserver DEMO
         (vserver services name-service cache netgroups ip-to-netgroup show)
Vserver IP Address Netgroup Source Create Time
------------------------
DEMO 10.193.67.222 netgroup1 ldap 2/21/2020 09:10:27
```

When a qtree is queried, it queries the entire path:

```bash
cluster:~$ export-policy check-access -vserver DEMO -volume netgrpvol -client-ip 10.193.67.222 -authentication-method sys -protocol nfs3 -access-type read-write -qtree tree
Path Policy Policy Rule
------------------------
/netgrpvol default vsroot volume 2 read
```

How to Configure LDAP in ONTAP © 2020 NetApp, Inc. All Rights Reserved.
When you change the export policy rule to deny writes, following is the result:

```
cluster::*> export-policy rule modify -vserver DEMO -policyname netgroup -ruleindex 1 -rorule never
```

```
cluster::*> export-policy check-access -vserver DEMO -volume netgrpvol -client-ip 10.193.67.222 -authentication-method sys -protocol nfs3 -access-type read-write
```

```
Path                  Policy     Owner    Owner Type  Index  Access
----------------------------------------------------------------
/                      default    vsroot    volume    2       read
/netgrpvol             netgroup   netgrpvol volume  1       denied
```

cache flush

Export policies also use caches to reduce the amount of time that is spent looking up IP addresses, host names, netgroups, and so on. This approach helps reduce the amount of time that is needed for mounting and traversing NFS exports.

In some cases, caches might become stale and require flushing, such as when a netgroup is changed to remove or to add a client, or if a client’s access level has changed. You can view caches through the other export-policy commands that are listed in this section.

The following command can flush individual caches or all caches en masse:

```
cluster::*> export-policy cache flush ?
[-vserver] <vserver name>      Vserver
[[<node> <nodename>]]          Node
[-cache {all|access|host|id|name|netgroup|showmount|ip}]  Cache Name
```

**Note:** To avoid having to repopulate all caches, when possible, flush only individual caches or use other commands in this section to flush individual entries.

The following caches are available to help you flush caches:

```
cluster::*> export-policy cache flush -vserver DEMO -cache ?
all             All
access          Access Cache in the Nblade
host            Host Name to IP Cache in the Mgwd
id              ID to Credential Cache in the Mgwd
name            Name to ID Cache in the Mgwd
netgroup        Netgroup cache in the Mgwd
showmount       Showmount Caches in the Mgwd and the Nblade
ip              IP to Host Name Cache in the Mgwd
```

access-cache

With the access-cache command, you can view and manage the export policy access caches. With this command, you can specify policies and hosts that have accessed a volume through NFS.

In the following instance, two clients attempt to access a volume with the netgroup export policy. Only one has access:

```
cluster::*> export-policy check-access -vserver DEMO -volume netgpvol -client-ip 10.193.67.225 -authentication-method sys -protocol nfs3 -access-type read-write
```

```
Path                  Policy     Owner    Owner Type  Index  Access
----------------------------------------------------------------
/                      default    vsroot    volume    2       read
/netgrpvol             netgroup   netgpvol volume  0       denied
```

```
cluster::*> export-policy check-access -vserver DEMO -volume netgpvol -client-ip 10.193.67.222 -authentication-method sys -protocol nfs3 -access-type read-write
```

```
Policy     Policy     Rule
---------- ---------- ----
```
Path                          Policy     Owner   Owner Type  Index Access
---------------------------------------- --------------------- ------- --------
/                             default    vsroot   volume   2 read
/netgrpvol                    netgroup   netgrpvol volume   1 read

When the .225 client attempts to mount, it is denied.

[root@centos7 ~]# mount -o nfsvers=3 DEMO:/netgrpvol /netgrpvol
mount.nfs: access denied by server while mounting DEMO:/netgrpvol

When the .222 client mounts, it is allowed.

[root@centos8-ipa ~]# mount -o nfsvers=3 DEMO:/netgrpvol /netgrpvol
[root@centos8-ipa ~]# mount | grep netgrp
DEMO:/netgrpvol on /netgrpvol type nfs (rw,relatime,vers=3,rsize=1048576,wsize=1048576,
namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,mountaddr=10.193.67.219,mountvers=3,mountport=635,mountproto=udp,local_lock=none,addr=10.193.67.219)

This result is reflected in the access-cache. Positive denotes successful access, and negative denotes unsuccessful access. Caches are per node and depend on nodes that own data LIFs where clients are connected.

cluster::*> export-policy access-cache show -node node2 -vserver DEMO -policy netgroup -address 10.193.67.222 -instance

          Node: node2
          Vserver: DEMO
          Policy Name: netgroup
          IP Address: 10.193.67.222
          Access Cache Entry Flags: has-usable-data
          Result Code: 0
          First Unresolved Rule Index: -
          Unresolved Clientmatch: -
          Number of Matched Policy Rules: 1
          List of Matched Policy Rule Indexes: 1
          Age of Entry: 83s
          Access Cache Entry Polarity: positive
          Time Elapsed since Last Use for Access Check: 83s
          Time Elapsed since Last Update Attempt: 83s
          Result of Last Update Attempt: 0
          List of Client Match Strings: @netgroup1

cluster::*> export-policy access-cache show -node node2 -vserver DEMO -policy netgroup -address 10.193.67.225 -instance

          Node: node2
          Vserver: DEMO
          Policy Name: netgroup
          IP Address: 10.193.67.225
          Access Cache Entry Flags: has-usable-data
          Result Code: 0
          First Unresolved Rule Index: -
          Unresolved Clientmatch: -
          Number of Matched Policy Rules: 0
          List of Matched Policy Rule Indexes: -
          Age of Entry: 41s
          Access Cache Entry Polarity: negative
          Time Elapsed since Last Use for Access Check: 41s
          Time Elapsed since Last Update Attempt: 41s
          Result of Last Update Attempt: 0
          List of Client Match Strings: -

netgroup

The export-policy netgroup command has the following available options:
In ONTAP 9.3 and later, a global name service cache was added (see TR-4668 for details), so netgroup cache commands were moved to the `vserver services name-service netgroup` command. However, you can still check membership for netgroups by using `check-membership`. The success of this command depends on the existence of an export policy rule that contains the netgroup that is being queried.

```
cluster::*> export-policy rule show -vserver DEMO -policyname netgroup
Policy          Rule    Access   Client                RO
Vserver      Name            Index   Protocol Match   Rule
------------ -------------- ------ ---------- -------------- --------------
DEMO         netgroup        1       any      @netgroup1            any
```

```
cluster::*> export-policy netgroup check-membership -vserver DEMO -netgroup netgroup1 -client-ip 10.193.67.222
Client 10.193.67.222 is a member of netgroup "netgroup1" for Vserver "DEMO" with state "name service cache".
```

For more information about the `check-membership` command, see the manual pages.

**CIFS Domain**

Active Directory domains can trust other domains. If an LDAP user lives in a trusted domain, then ONTAP can query either domain with the same bind credentials. In some cases, the domain trust might not be working properly, so you might need to use the following commands to find an issue.

**cifs domain trusts**

This command reviews what ONTAP sees when it queries the domain trusts for the joined Active Directory domain, and it offers the option to rediscover the trusts. If a trusted domain does not show properly, LDAP communication between the domains does not work.

```
NAME
    vserver cifs domain trusts -- Manage discovered trusted domains

DESCRIPTION
    Manage discovered trusted domains

COMMANDS
    rediscover - Reset and rediscover trusted domains for a Vserver
    show - Display discovered trusted domain information
```

**cifs domain discovered-servers**

This command enables storage administrators to see how ONTAP sees the Active Directory domain to verify whether server connectivity to domain controllers is functioning properly.

```
cluster::*> cifs domain discovered-servers ?
    discovery-mode> *The discovery-mode directory
    reset-servers     Reset and rediscover servers for a Vserver
    show             Display discovered server information
```

**event log show**

When an issue occurs in ONTAP, it is logged in the event log subsystem. These logs are maintained for a period on-box and are available through NetApp Active IQ® digital advisor if you need to look at issues earlier in the event log history that might have rolled off.
You can filter event log messages by date, time, severity, message name, and various other ways, as in the following list of command options:

```
cluster::*> event log show ?
  [ -detail | -detailtime | -instance | -fields <fieldname>, ... ]
  [ [+node] <nodename> ]    Node
  [ [+seqnum] <Sequence Number> ]    Sequence#
  [ -time <"MM/DD/YYYY HH:MM:SS"> ]    Time
  [ -severity {EMERGENCY|ALERT|ERROR|NOTICE|INFORMATIONAL|DEBUG} ]
  Severity (default: <=ERROR)
  [ -ems-severity {NODE_FAULT|SVC_FAULT|NODE_ERROR|SVC_ERROR|WARNING|NOTICE|INFO|DEBUG|VAR} ]
  *EMS Severity
  [ -source <text> ]    Source
  [ -message-name <Message Name> ]    Message Name
  [ -event <text> ]    Event
  [ -kernel-generation-num <integer> ]    *Kernel Generation Number
  [ -kernel-sequence-num <integer> ]    *Kernel Sequence Number
  [ -action <text> ]    Corrective Action
  [ -description <text> ]    Description
  [ -filter-name <text> ]    Filter Name
```

**Note:** You can find more information about these commands in the manual pages for your ONTAP release.

For name service issues, such as LDAP connectivity, you can use a specific subset of message names to filter event logs to find issues faster. The following list is not comprehensive, but it can get you started.

**Note:** Message names are case-sensitive.

**For LDAP:**

```
cluster::*> event route show -message-name *ldap*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq Threshold</th>
<th>Time Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap.false.configs.removed</td>
<td>NOTICE</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.ldap.byhost.missing</td>
<td>INFORMATIONAL</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.ldap.config</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.connectFailure</td>
<td>ALERT</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.hostnames.not.resolved</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.hostnames.resolved.partially</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.noServers</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.query.timed.out</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.referralError</td>
<td>INFORMATIONAL</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdldap.slowServer</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secdnetgroup.ldap.badFilter</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**For exports:**

```
cluster::*> event route show -message-name *export*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq Threshold</th>
<th>Time Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nblade.exportAccessChkFailed</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Nblade.exportAccessIndeterm</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>crypto.export.failed</td>
<td>ALERT</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.anon.noCredForId</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.dns.config</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.dom.notFound</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.dom.transient</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.hostname.notFound</td>
<td>INFORMATIONAL</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.hostname.transient</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.dnsNoPtrRec</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.notFound</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.partial</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.ngbh.allFailed</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.anonNameToId</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.policy.empty</td>
<td>NOTICE</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
exports.policy.last.rule         ERROR - 0 0

For NFS authorization:

```
cluster::*> event route show -message-name *nfsAuth*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>secd.nfsAuth.noCifsCred</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nfsAuth.noCifsSId</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nfsAuth.noCifsUser</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nfsAuth.noNameMap</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nfsAuth.noUnixCreds</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nfsAuth.problem</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

```
cluster::*> event route show -message-name *unix*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>secd_unixLookupFailure</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

For netgroups:

```
cluster::*> event route show -message-name *netgroup*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>exports.netgroup.dnsNoPtrRec</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.notFound</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>exports.netgroup.partial</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.files.missing</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.ldap.byhost.missing</td>
<td>INFORMATIONAL</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.ldap.config</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.nis.byhost.decode</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.nis.byhost.missing</td>
<td>INFORMATIONAL</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>netgroup.nis.config</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.netgroup.ldap.badFilter</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

For name translations:

```
cluster::*> event route show -message-name *nameTrans*
```

<table>
<thead>
<tr>
<th>Message</th>
<th>Severity</th>
<th>Destinations</th>
<th>Freq</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>secd.nameTrans.groupNotFound</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nameTrans.invalidConfig</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nameTrans.invalidUser</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nameTrans.noNameMapping</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>secd.nameTrans.unknownUser</td>
<td>ERROR</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics

ONTAP has a statistics subsystem that you can enable to track down performance issues or to look for incrementing errors.

These statistics are on demand and are started with the `statistics start` command, with filters available for objects and counters.

Following is the manual page entry:

```
cluster::*> man statistics start
```

<table>
<thead>
<tr>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>statistics start -- Start data collection for a sample</td>
</tr>
</tbody>
</table>
This command is available to cluster and Vserver administrators at the advanced privilege level.

DESCRIPTION
This command starts the collection of performance data. Use the statistics stop command to stop the collection. You view the sample of performance data by using the statistics show command. You can collect more than one sample at a time.

PARAMETERS
[-object <text>] - Object
Selects the objects for which you want to collect performance data. This parameter is required. To view a list of valid object names, type statistics catalog object show at the command prompt. To specify multiple objects, use "|" between each object.

Caution: You should limit the scope of this command to only a few objects at a time to avoid a potentially significant impact on the performance of the system.

[-instance <text>] - Instance
Selects the instances for which you want to collect performance data. If you do not specify this parameter, the command collects statistics for all of the instances associated with the specified objects. To specify multiple instances, use "|" between each instance.

For example, if you want to collect disk object statistics, you can use this parameter to specify the name of a specific disk whose statistics you want to view. If you do not specify this parameter, the command will collect statistics for all disks in the system.

[-counter <text>] - Counter
Selects the counters for which you want to collect performance data. If you do not specify this parameter, the command collects statistics for all of the counters in the specified objects. To specify multiple counters, use "|" between each counter.

[-preset <text>] - Preset
If this parameter is specified, the command displays statistics for the specified preset.

[-sample-id <text>] - Sample Identifier
Specifies an identifier for the sample. Identifiers must be unique and are restricted to the characters 0-9, a-z, A-Z, and ".". If you do not specify this parameter, the command generates a sample identifier for you and defines this sample as the default sample for the CLI session. When you run the statistics show command without specifying the -sample-id parameter, data from the default sample displays. If you run this command during the same CLI session and do not specify the -sample-id parameter, the command overwrites the previous sample. The command does not delete the default sample when you close your session.

[-vserver <vservename>] - Vserver
Selects the vserver for which you want to collect performance data. If you do not specify this parameter, the command collects statistics for all of the Vservers in the cluster.

[-node {<nodename>|local}] - Node
Selects the node for which you want to collect performance data. If you do not specify this parameter, the command collects statistics for all of the nodes in the cluster.

[-filter <text>] - Filter
Selects performance data for the instance that matches the specified filter criteria. For example, to display the instances from node1, specify -filter "node_name=node1".

[-duration <integer>] - Sample Duration in Minutes
If this parameter is specified, the command will collect the closing sample after the time specified. Duration can be specified in minutes.

[-sample-type {User|System}] - Sample Type (privilege: diagnostic)
If this parameter is specified, the command will set the sample owner type to user or system. The default sample type is user.

[-max <integer>] - Tracker Size
Specifies the number of most active instances of an active object to display. The default setting is to display all of the instances.

[-sort-key <text>] - Counter Used For Sorting
If this parameter is specified, the command displays statistics sorted by the specified counter. Only one counter can be specified.
[-sort-order {ascending|descending}] - Sort Order
This parameter may be used in conjunction with the -sort-key parameter. This parameter changes the order in which statistics are sorted. Possible values are ascending and descending. The default setting is descending.

For name services and LDAP statistics, you can specify the following objects with the -object option, which can help you with potential issues and troubleshooting:

accesscache*
credstore
external_service*
nfs_credential
nfs_exports_access_cache
nfs_exports_cache
secd*

You can specify multiple objects in a single statistics command with the pipe (|) value. For example:

cluster::*> statistics start -object
accesscache*|credstore|external_service*|nfs_credential|secd*nfs_exports_access_cache|nfs_exports_cache
Statistics collection is being started for sample-id: sample_235

Statistics for caches give information about the number of cache entries, how many times the cache was hit, how many missed cache entries there were, and so on.

Statistics for external services give information about DNS and LDAP queries, such as latency, number of operations, and servers that are being used.

In the following example, you can filter by Vserver/SVM, object, and instance name to obtain a refined view of the external service counters—all the way down to the LDAP server IP address. The use case here is to find out how many anonymous binds have occurred from this SVM.

cluster::*> statistics show -vserver NFS -object external_service* -instance
*LDAP*AnonymousBind:10.193.67.222 -counter num*

Object: external_service_op
Start-time: 3/3/2020 15:16:43
Elapsed-time: 529s
Scope: NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_not_found_responses</td>
<td>0</td>
</tr>
<tr>
<td>num_request_failures</td>
<td>0</td>
</tr>
<tr>
<td>num_requests_sent</td>
<td>0</td>
</tr>
<tr>
<td>num_responses_received</td>
<td>0</td>
</tr>
<tr>
<td>num_successful_responses</td>
<td>0</td>
</tr>
<tr>
<td>num_timeouts</td>
<td>0</td>
</tr>
</tbody>
</table>

The next example shows how to find out how many times a user has been looked up.

cluster::*> statistics show -vserver NFS -object external_service* -instance
*LDAP*GetUserInfoFromName:10.193.67.222 -counter num*

Object: external_service_op
Instance: NFS:LDAP (NIS & Name Mapping):GetUserInfoFromName:10.193.67.222
Start-time: 3/3/2020 15:16:43
End-time: 3/3/2020 15:27:40
Elapsed-time: 657s
Scope: NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_not_found_responses</td>
<td>0</td>
</tr>
<tr>
<td>num_request_failures</td>
<td>1</td>
</tr>
</tbody>
</table>
For `secd*` objects, you can find out the number of times that an error has occurred since the statistics were started. The specific object is `secd_rpc_error`:

```
cluster::*> statistics show -vserver NFS -object secd_rpc_error -instance *ERROR* -counter count
```

### Object: secd_rpc_error
- **Instance:** NFS:secd_rpc_check_ldap_config:RESULT_ERROR_SECD_NO_SERVER_AVAILABLE
- **Start-time:** 3/3/2020 15:16:43
- **End-time:** 3/3/2020 15:31:49
- **Elapsed-time:** 906s
- **Scope:** NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>0</td>
</tr>
<tr>
<td>count</td>
<td>0</td>
</tr>
</tbody>
</table>

### Object: secd_rpc_error
- **Instance:** NFS:secd_rpc_ldap_get_netgroup_match_by_host:RESULT_ERROR_SECD_GROUP_NOT_FOUND
- **Start-time:** 3/3/2020 15:16:43
- **End-time:** 3/3/2020 15:31:49
- **Elapsed-time:** 906s
- **Scope:** NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1</td>
</tr>
</tbody>
</table>

### Object: secd_rpc_error
- **Instance:** NFS:secd_rpc_ldap_get_netgroup_match_by_host:RESULT_ERROR_SECD_NETGROUP_BYHOST_NOT_ENABLED
- **Start-time:** 3/3/2020 15:16:43
- **End-time:** 3/3/2020 15:31:49
- **Elapsed-time:** 906s
- **Scope:** NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>0</td>
</tr>
</tbody>
</table>

### Object: secd_rpc_error
- **Instance:** NFS:secd_rpc_ldap_get_passwd:RESULT_ERROR_SECD_USER_NOT_FOUND
- **Start-time:** 3/3/2020 15:16:43
- **End-time:** 3/3/2020 15:31:49
- **Elapsed-time:** 906s
- **Scope:** NFS

<table>
<thead>
<tr>
<th>Counter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>0</td>
</tr>
<tr>
<td>count</td>
<td>0</td>
</tr>
</tbody>
</table>

6 entries were displayed.

After the statistics have run for as long as you want them to run, you should turn them off to reduce load on the cluster and to keep statistics files small in size.

```
cluster::*> statistics stop
Statistics collection is being stopped for sample-id: sample_235
```
You can also view or delete existing statistics samples.

```
cluster::*> statistics samples
delete show
```

ds vserver security file-directory show

For storage administrators to easily view the permissions of files and folders in a file system, ONTAP provides the command `vserver security file-directory show`. This command is useful for troubleshooting permissions or access issues.

To run this command, use the SVM and full path of the file or folder in the volume. For more information about the command, use `man vserver security file-directory show`.

The following examples show the output from the command based on the security style of the object and the style of ACL that is applied.

**Example of NTFS security style:**

```
cluster::> vserver security file-directory show -vserver DEMO -path /data/Windows.iso

Vserver: DEMO
File Path: /data/Windows.iso
File Inode Number: 15770
Security Style: ntfs
Effective Style: ntfs
DOS Attributes: 20
DOS Attributes in Text: ---A----
Expanded Dos Attributes:
  UNIX User Id: 0
  UNIX Group Id: 0
  UNIX Mode Bits: 777
UNIX Mode Bits in Text: rwxrwxrwx
ACLs: NTFS Security Descriptor
  Control:0x8004
  Owner:BUILTIN\Administrators
  Group:NTAP\DomainUsers
  DACL - ACEs
    ALLOW= Everyone=0x1f01ff-(Inherited)
```

**Example of UNIX security style—mode bit permissions:**

```
cluster::> vserver security file-directory show -vserver DEMO -path /flexgroup_16/newfile1

Vserver: DEMO
File Path: /flexgroup_16/newfile1
File Inode Number: 3358476
Security Style: unix
Effective Style: unix
DOS Attributes: 20
DOS Attributes in Text: ---A----
Expanded Dos Attributes:
  UNIX User Id: 0
  UNIX Group Id: 0
  UNIX Mode Bits: 644
UNIX Mode Bits in Text: rw-r--r--
ACLs: -
```

**Example of mixed security style—NFSv4.x ACLs:**

```
cluster::> vserver security file-directory show -vserver DEMO -path /home/student2

Vserver: DEMO
File Path: /home/student2
File Inode Number: 97
Security Style: mixed
Effective Style: unix
```
Security styles affect how name mappings occur in ONTAP, and the user who attempts access must resolve to a user name that fits into the permission structures that are set on the object.

5.4 What Information to Collect Before You Contact NetApp Support

If you encounter an LDAP issue and cannot resolve it on your own, NetApp Support is readily available to assist. When you open a support case, the technical support engineer must gather some data to troubleshoot the issue. To expedite that process, the following is a list of questions that you can answer and information that you can provide to help resolve support cases faster. This list is not exhaustive—you might be asked for more data from the technical support engineer—but it is a start:

- What date and time did the problem occur?
- What LDAP server type and OS are being used?
- What LDAP clients are being used?
- Which user or group is affected?
- Is the problem still occurring? Is it intermittent?
- Can the LDAP administrator be present for the call in case the technical support engineer needs extra information from the LDAP server?
- Does the LDAP server use any sort of encryption?
- Does the issue occur on all nodes? Some nodes? Specific IP addresses?
• Can the LDAP server be reached through the network from ONTAP?
• Generate a new AutoSupport report with `autosupport invoke * -type all`:
  – This command gathers information about the LDAP client, client schema, DNS, network, event logs, and so on.
• Have you checked the troubleshooting command output by using the instructions in the “ONTAP CLI Commands for LDAP Troubleshooting” section?
• Can other clients outside of ONTAP query LDAP properly with the commands that are listed in the “Third-Party Tools and Utilities” section?
• For LDAP query issues (unable to find user) or LDAP configuration issues, collect output from the LDAP schema for a user and a group. This output helps verify that the correct schemas are being used.
• Packet traces during an issue from the client, LDAP server, and ONTAP system.
• Provide statistics capture information for name services and caches; see the “Statistics” section.

For information about collecting packet traces in ONTAP, see the following NetApp Knowledge Base articles:

- [How to Capture Packet Traces (tcpdump) on ONTAP 9.2+ Systems](#)
- [How to Collect Rolling Packet Traces Using pktt on ONTAP 9.1 and Below Systems](#)

### 6 Best Practices

This section covers best practices for specific scenarios when you use LDAP with NetApp ONTAP. These best practices are not hard requirements; they are simply guidelines for success. Best practices can help prevent simple issues, but they are not guarantees that no issues will occur in an environment. This list is not exhaustive, but it attempts to address multiple scenarios.

#### 6.1 LDAP Server Best Practices

Following is a summary of best practices for LDAP servers:

- Use multiple LDAP servers to prevent overloading a single server.
- Keep LDAP servers updated to the latest OS release and patch release.
- Where possible, use encryption for binds and LDAP searches (such as SASL bind and StartTLS or SMB signing for queries).
- Use `StartTLS` instead of LDAP over SSL (LDAPS); LDAPS has been deprecated as a standard.
- Create service accounts to use with binds to LDAP clients in ONTAP rather than the administrator or Directory Manager users.
- Disable anonymous binds on the LDAP server.
- In Windows Active Directory LDAP where multiple domains in a forest have UNIX attributes, consider replicating attributes to the global catalog for UNIX identity management. See section 4.3, “LDAP That Uses Active Directory for UNIX Identity Management” and the “Active Directory Global Catalog Searches” subsection.
- Create DNS A records for multiple LDAP servers by using the same name (for example, `ldap.ntap.local`) to provide DNS load balancing or use an external network load balancer to help spread network connections across servers.
- If possible, set up automation for user and group creation that autopopulates UNIX attributes in the LDAP server. This best practice pertains mainly to multiprotocol NAS environments, where Windows and UNIX accounts are being used.
- When you create UNIX user names in multiprotocol NAS environments, to avoid the need for explicit name mapping rules, try to use the same user name as the Windows name.
6.2 DNS Server Best Practices

Following is a summary of best practices for DNS servers. TR-4523 covers DNS load balancing in detail:

- For redundancy and load balancing, use multiple DNS servers that replicate information to each other.
- Ensure that hosts have forward- and reverse-lookup records in DNS.
- Verify that LDAP has SRV records available in DNS.
- For redundancy, specify multiple DNS servers in the ONTAP SVM DNS configuration.

6.3 Cache Management Best Practices

This section covers a summary of best practices for cache management in ONTAP.

Cache Timeout Settings

Name service caches have specific timeout settings by default, because caches should age out over time to prevent situations in which entries become stale. The default cache settings in ONTAP 9.7 are listed in Table 13.

Many cache timeout values in ONTAP are modifiable. The following best practices apply to timeout settings for caches:

- In nearly all cases, leave the cache timeouts as the defaults. To remove stale entries in caches, use `cache flush` commands.
- If your environment has high churn (users are constantly added or removed from groups, netgroups add or remove hosts, and so on) consider lowering the cache timeout values to update more frequently.
- If your environment has low churn (users rarely change and netgroups rarely change), consider increasing the timeout values to maintain caches longer.
- Keep in mind that lowering the cache timeout values increases network traffic between ONTAP and name services and increases the load on CPU for ONTAP and name services, which negatively affect performance. Increasing the cache timeout values results in more frequent stale entries, and users might be incorrectly granted or denied access.

Manual Cache Flush Considerations

By flushing caches manually, you remove information from them that might be outdated because of the following reasons:

- A recent change to export policy rules
- A recent change to user information or group membership
- A recent change to host name records in name servers
- A recent change to netgroup entries in name servers
- Recovery from a network outage that prevented netgroups from being fully loaded

Flushing the caches removes the outdated information and forces ONTAP to retrieve the current information from the appropriate external resources.

**Note:** Repopulation of caches can be resource intensive. You should flush a cache only if you are trying to resolve a specific issue, and if possible, flush only the offending entry.

Some caches can have individual entries removed. Other caches must be flushed en masse. Table 14 shows at what level caches can be flushed in ONTAP 9.7 and later. The level to which a cache can be flushed also often corresponds to the level that it can be viewed.
Table 14) Manual cache flush operability in ONTAP 9.7 and later.

<table>
<thead>
<tr>
<th>Cache</th>
<th>Cache Flush Operability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Netgroup member cache</td>
<td>Individual clients or all clients</td>
</tr>
<tr>
<td>Netgroup ip-to-host cache</td>
<td>Individual clients or all clients</td>
</tr>
<tr>
<td>LDAP group authentication</td>
<td>Individual names</td>
</tr>
<tr>
<td>Export-policy cache</td>
<td>Individual caches or all caches</td>
</tr>
<tr>
<td>Name service: group membership</td>
<td>Individual user and group</td>
</tr>
<tr>
<td>Name service: hosts</td>
<td>Individual forward and reverse DNS lookup</td>
</tr>
<tr>
<td>Name service: UNIX users and groups</td>
<td>Individual users and groups or all users and groups</td>
</tr>
<tr>
<td>Export-policy access-cache</td>
<td>Individual client/policy</td>
</tr>
</tbody>
</table>

6.4 Export Policy Best Practices

This section covers a summary of best practices for export policies and rules in ONTAP.

Export policies and rules help control access to NFS exports based on the client that is requesting access. You can find most general export policy best practices in TR-4067. The following best practices pertain to export policies and rules as they relate to LDAP servers:

- If you specify a large number of clients with the same access permissions in an export policy, to reduce the complexity of storage management, consider using netgroups.
- Ensure that the SVM’s DNS configuration is working properly and can resolve hosts through forward and reverse lookups.
- With export policy rules, the rule index order determines which export policy rule is applied first. If you want to set a more restrictive policy to an entire subnet but allow specific clients to have root access to a mount, set the rule index for the admin clients higher in the list so that they are processed first. If the more restrictive policy rule is listed first, then clients are denied access based on that policy, even if root access is granted later in the rule index.
- If you specify a netgroup in an export policy rule, use the @netgroup syntax; otherwise, ONTAP interprets the netgroup as a host name. For example, if you specify netgroup1 in a rule, use @netgroup1.

6.5 Netgroup Best Practices

Following is a summary of best practices for netgroups in ONTAP:

- Ensure that hosts added to netgroups exist in DNS.
- Specify netgroups in export policy rules by using the @ sign.
- When you use netgroup.byhost, consider the following to enable the desired access results for hosts:
  - Forward and reverse DNS records for host names.
  - Host triple entry in netgroup file.
  - Netgroup specification for the host’s netgroup.byhost entry.
  - Need to always use lowercase hosts to avoid case-sensitivity issues.
  - Syncing of DNS and netgroup.byhost entries, including case sensitivity.
  - If you use netgroup.byhost with NIS, be sure that the triples are configured to avoid using dashes (–) in the entries. For example, the entries should look like (host,,) not (host,–,–).
ONTAP supports only the host portion of the triple. NIS treats any entry in the other portions of the triple as an attempted entry.

- NetApp highly recommends that you use `netgroup.byhost` functionality for large environments with very large netgroups. To enable access to work properly, the `netgroup.byhost` and `netgroup` entries must be synchronized.

### 6.6 LDAP Client Best Practices

Following is a summary of best practices for LDAP clients (including ONTAP, which is also an LDAP client):

- Configure LDAP clients to bind and to search with the most secure methods available. For ONTAP, it includes binding as the CIFS server (if it exists), binding with SMB signing, and/or using LDAPS or StartTLS.
- For Linux clients, NetApp recommends SSSD for its ease of configuration and setup.
- When you use SSSD for LDAP, only the LDAP identity provider is supported. The Active Directory identity provider generates unique UID numerics based on the Active Directory SIDs and is not supported in ONTAP today. For more information, see the section “SSSD UID/GID Algorithms (SSSD Active Directory Provider).”
- Ensure that LDAP clients all return the same user and group information (including group memberships) through `ldapsearch` and ONTAP CLI commands.
- Avoid having local users or groups on the clients that have conflicting names or numeric IDs. This situation can cause unpredictable behavior for file and folder access.

### Appendix: Command Examples and Other Information

This section presents command examples and other information that did not quite fit in the previous sections of this document.

#### Use of Local Files in ONTAP

In some cases, it might not be possible (no access to external name services) or necessary (not enough users to justify setting up LDAP) to use name service servers for users, groups, netgroups, and so on.

In those cases, NetApp ONTAP provides several methods for you to create local file entries for these objects. It is still a best practice to use external name services because of the centralized management and consistency across clients.

There are two main ways to use local files in ONTAP:

- Create individual entries for users, groups, hosts, and so on, through System Manager or the CLI.
- Import files into the ONTAP SVM through the `load-from-uri` command, which requires access to HTTP or FTP servers that can pull the files over.

You can use local files for users, groups, group memberships, hosts, NIS databases, and netgroups.

For more information about how to create local users and groups, see the product documentation.

#### Scaled Mode/File-Only Mode

In some cases, an environment might have thousands of users and groups, but no LDAP server. However, by default, there are limits on the number of local users and groups that you can create per SVM.

Scaled mode/file-only mode for local users and groups in ONTAP 9.1 enables storage administrators to expand the limits of local users and groups. It does so by enabling a `diag-level` name service option and
then using the load-from-uri functionality to load files into the cluster to provide higher numbers of users and groups. Scaled mode/file-only mode also can improve the performance of name service lookups, because there is no longer a need for external dependencies on name service servers, networks, and so on. However, this performance comes at the expense of ease of management of the name services, because file management adds overhead to the storage management and introduces more potential for human error. In addition, local file management must be performed per cluster, adding an extra layer of complexity.

To enable this option for users and groups, use the `vserver services name-service unix-user file-only` and `vserver services name-service unix-group file-only` commands.

<table>
<thead>
<tr>
<th>NAME</th>
<th>vserver services name-service unix-user file-only modify -- Change configuration for UNIX-user file download</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVAILABILITY</td>
<td>This command is available to cluster and Vserver administrators at the diagnostic privilege level.</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>The vserver services name-service unix-user file-only modify command enables you to load UNIX user files with large number of UNIX users beyond the maximum configurable limit of 65536 for the cluster. Once it is enabled, individual operations on UNIX users are not allowed, and the users can only be managed using the vserver services name-service unix-user load-from-uri command.</td>
</tr>
</tbody>
</table>
| PARAMETERS | `-vserver <vserver name>` - Vserver Use this parameter to specify the Vserver for which you want to modify the file-only mode. 
`[-is-enabled {true|false}]` - Is File-Only Download Enabled? Use this parameter with value true to enable the file-only mode. This field is set to false by default. |

After the mode is enabled, use the following command to load the user and group files from the URI.

```
cluster::*> vserver services name-service unix-user load-from-uri
```

**Note:** If you load files larger than 10MB for users and 25MB for groups, use the `-skip-file-size-check` option.

When you use file-only mode, individual operations on users and groups are not allowed. This configuration is not currently supported in NetApp MetroCluster™ or SVM disaster recovery (SVM DR) scenarios.

When you use this command, some warnings are issued:

```
cluster::*> vserver services name-service unix-user file-only modify -vserver SVM1 -is-enabled true
```

Warning: Do not enable the file-only configuration if you are using, or plan to use, MetroCluster or Vserver Async DR.
- If you enable the file-only configuration:
  - Modifying individual user entries will not be possible.
  - Local Unix-users must be managed by downloading a file using the "vserver services name-service unix-user load-from-uri" command.
  - Downloading the users will replace all existing users. The standard set of users must be present in the file. If the users "root", "pcuser" and "nobody" are not defined, a data serving interruption might occur.
  - This command may take some time to complete.

Do you want to continue? {y|n}: y

To check the status of local user and group files, use:

```
cluster::*> vserver services unix-user file status
```

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Node</th>
<th>Load Time</th>
<th>Hash Value</th>
</tr>
</thead>
</table>

© 2020 NetApp, Inc. All Rights Reserved.
Can You Still Use External Name Services?

File-only mode does not mean that you cannot use LDAP or NIS as a name service; it means that management of local users and groups is performed with files only. In the following example, file-only mode is enabled on the SVM, and LDAP can still be used to perform name lookups:

```
cluster::*=> name-service ns-switch show -vserver SVM1
                (vserver services name-service ns-switch show)

          Source
Vserver     Database    Order
----------   ----------   ----------
  SVM1       hosts       files, dns
  SVM1       group       files, ldap
  SVM1       passwd      files, ldap
  SVM1       netgroup    files
  SVM1       namemap     files
5 entries were displayed.

cluster::*=> vserver services unix-user file-only show -vserver SVM1

Vserver: SVM1
Is File-Only Download Enabled?: true

cluster::*=> getxxbyyyy getpwbyname -node ontap9-tme-8040-01 -vserver SVM1 -username ldapuser
            -show-source true
                (vserver services name-service getxxbyyyy getpwbyname)
Source used for lookup: LDAP
pw_name: ldapuser
pw_uid: 1108
pw_gid: 513
pw_gecos: 
pw_dir: /home/ldapuser
pw_shell: /bin/sh

Keep in mind that when file-only is enabled, the default local users of root, pcuser, and nobody are removed if the file that is being loaded does not have those users. Be sure to include the local users and groups in your passwd and group files when you use file-only.

cluster::*=> unix-user show -vserver SVM1

Error: show failed: File-only configuration is enabled. Use the command "vserver services name-service unix-user file show" instead.

cluster::*=> vserver services name-service unix-user file show -vserver SVM1

Line No  File content
----------  ----------------
  1  nobody:*:65535:65535::::
  2  pcuser:*:65534:65534::::
  3  root:*:0:1:::
Limits

This section covers the limits for local users and groups in ONTAP. Table 15 presents a summary. These limits are clusterwide.

Table 15) Limits on local users and groups in ONTAP clusters.

<table>
<thead>
<tr>
<th>Local UNIX User/Group Limits</th>
<th>File-Only User and Group Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>32,768 (default)</td>
<td>passwd file size (users): 10MB</td>
</tr>
<tr>
<td>65,536 (maximum)</td>
<td>group file size: 25MB</td>
</tr>
<tr>
<td></td>
<td>Note: You can override passwd and group file sizes with –skip-file-size-check, but larger file sizes have not been tested.</td>
</tr>
<tr>
<td></td>
<td>Users: 400,000</td>
</tr>
<tr>
<td></td>
<td>Groups: 15,000</td>
</tr>
<tr>
<td></td>
<td>Group memberships: 300,000</td>
</tr>
<tr>
<td></td>
<td>SVMs: 6</td>
</tr>
</tbody>
</table>

The local UNIX user and group limits are clusterwide and affect clusters with multiple SVMs. Thus, if a cluster has four SVMs, then the maximum number of users in each SVM must add up to the maximum limit set on the cluster.

For example:

- SVM1 has 2,000 local UNIX users.
- SVM2 has 40,000 local UNIX users.
- SVM3 has 20 local UNIX users.
- SVM4 then has 23,516 local UNIX users available to be created.

Any attempted creation of a UNIX user or group beyond the limit results in an error message.

Example:

```
cluster::> unix-group create -vserver NAS -name test -id 12345
Error: command failed: Failed to add "test" because the system limit of {limit number} "local unix groups and members" has been reached.
```

The limits are controlled by the following commands in the advanced privilege level:

```
cluster::*> unix-user max-limit
           modify show
```

Default Local Users

When you create an SVM by using vserver setup or System Manager, default local UNIX users and groups are created, along with default UIDs and GIDs.

The following shows these users and groups:

```
cluster::> vserver services unix-user show -vserver vs0
Vserver User User Group Full
---------- -------------- ---- ----
nfs nobody 65535 65535 -
nfs pcuser 65534 65534 -
nfs root 0 0 -
cluster::> vserver services unix-group show -vserver vs0
Vserver Name ID
---------- ----
nfs nobody 65535
```

123 How to Configure LDAP in ONTAP © 2020 NetApp, Inc. All Rights Reserved.
When you use file-only mode, be sure that the preceding users exist in the files that are being used to manage the cluster. After file-only is enabled, the default users are removed if the uploaded file does not include them.

**diag secd Versus vserver security access-check**

Secd refers to the user authentication application that resides in the system shell of ONTAP. This application has interacted with domain controllers, LDAP servers, DNS, and so on, since the early days of ONTAP running in cluster-mode. Throughout the years and ONTAP releases, the role of secd has changed a bit, and more operations have been moved to other areas; for example, LDAP functionality was moved to the FreeBSD libc modules.

NetApp has made these changes so that name service caches can be retained at an SVM level and become a global cache that is used regardless of the node in the cluster where a request arrived. Secd historically used localized caches on each node in versions earlier than ONTAP 9.3, which sometimes left cache differences across a cluster. For more information about global name service caches, see TR-4668: Name Services Best Practices Guide.

In addition to the global cache, new commands were created to accomplish several goals, including to:

- Simplify name service queries.
- Provide a command that did not require diag privileges to run (but could still use node-level queries as needed for troubleshooting).
- Remove the node requirement for running the command.

In ONTAP 9.6 and later, a new set of commands was introduced that mimics the name lookup capabilities of **diag secd commands.** When possible, you should use these commands instead of **diag secd.**

```
cluster::*> vserver services access-check ?
  authentication>             *Check Authentication Information
  dns>                        *Check DNS Lookups
  name-mapping>               *Check Name Mapping Operations
  server-discovery>           *Check Server Discovery Information
```

**LDAP Query Examples**

The following examples show what LDAP queries look like.

**LDAP User Attribute Dump Example: Active Directory**

A PowerShell query returns a series of attribute values that you can use to figure out the appropriate LDAP schema template to apply to your configuration. The following example shows a user dump from Active Directory LDAP with filters applied to `-Properties` to make it easier to find the attributes that are needed.

```
PS C:\Users\Administrator> Get-ADUser -Identity prof1 -Properties Name,gecos,gidNumber,HomeDirectory,ObjectClass,sAMAccountName,uid,uidNumber,unixHomeDirectory
DistinguishedName : CN=prof1,CN=Users,DC=NTAP,DC=local
Enabled : True
gecos : Prof1
gidNumber : 1101
GivenName : prof1
HomeDirectory :
Name : prof1
```
ObjectClass       : user
ObjectGUID        : 0973b3f1-da85-499c-80b4-a210e0d0fb2f
SamAccountName    : prof1
SID               : S-1-5-21-3552729481-4032800560-2279794651-1110
Surname           :
uid               : (prof1)
uidNumber         : 1100
unixHomeDirectory : /home/prof1
UserPrincipalName : prof1@ONTAP_LOCAL

Keep in mind that PowerShell queries work only with versions of Windows that support them. For older Windows versions or for Linux/UNIX-based LDAP servers, you can use `ldapsearch` commands.

**LDAP User Attribute Dump Example: ldapsearch**

For common `ldapsearch` examples, see [EXAMPLES OF COMMON LDAPSEARCHES](#).

The following output is from an `ldapsearch` query:

```
# ldapsearch -D "cn=Directory Manager" -w -p 389 -h 10.193.67.222 -b "dc=centos-ldap,dc=local" -s
# sub "(uid=ipa-user)"
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <dc=centos-ldap,dc=local> with scope subtree
# filter: (uid=ipa-user)
# requesting: ALL
#
# ipa-user, users, compat, centos-ldap.local
dn: uid=ipa-user,cn=users,cn=compat,dc=centos-ldap,dc=local
objectClass: posixAccount
objectClass: ipaOverrideTarget
objectClass: top
gecos: IPA User
cn: IPA User
uidNumber: 1971600003
gidNumber: 1971600000
loginShell: /bin/sh
homeDirectory: /home/ipa-user
ipaAnchorUUID:: OklQQTpjZW50b3MtbGRhcC5sb2NhbDplMDQwZWU0Mi00ODRjLTExZWEtOWE3ZC0wMDUwNTY5ZWM0N2Q=
uid: ipa-user

# ipa-user, users, accounts, centos-ldap.local
dn: uid=ipa-user,cn=users,cn=accounts,dc=centos-ldap,dc=local
givenName: IPA
sn: User
uid: ipa-user
cn: IPA User
displayName: IPA User
initials: IU
gecos: IPA User
krbPrincipalName: ipa-user@CENTOS-LDAP.LOCAL
gidNumber: 1971600000
userClass: user
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
objectClass: inetuser
objectClass: posixaccount
objectClass: krbprincipalaux
objectClass: krbticketpolicyaux
objectClass: ipaobject
objectClass: ipasshuser
objectClass: ipaSshGroupOfPubKeys
```
Sample Credential Dump

The following command fetches all the information about a user from the configured name services and protocol services. This command is especially valuable when you are troubleshooting name mapping and permissions issues in ONTAP in multiprotocol environments. Keep in mind that this command works only when CIFS/SMB is configured, because there must be Active Directory communication to fetch Windows credentials.

```
cluster::*> diag secd authentication show-creds -node ontap9-tme-8040-01 -vauser DEMO -unix- user-name prof1 -list-id true -list-name true

UNIX UID: 1100 (prof1) <> Windows User: S-1-5-21-3552729481-4032800560-2279794651-1100
   (NTAP\prof1 (Windows Domain User))

GID: 1101 (ProfGroup)
Supplementary GIDs:
   1101   (ProfGroup)
   1201   (group1)
   1202   (group2)
   1203   (group3)
   1220   (sharedgroup)

Primary Group SID: S-1-5-21-3552729481-4032800560-2279794651-111 NTAP\DomainUsers (Windows Domain group)

Windows Membership:
   S-1-5-21-3552729481-4032800560-2279794651-1106 NTAP\group2 (Windows Domain group)
   S-1-5-21-3552729481-4032800560-2279794651-1107 NTAP\DomainUsers (Windows Domain group)
   S-1-5-21-3552729481-4032800560-2279794651-1122 NTAP\sharedgroup (Windows Domain group)
   S-1-5-21-3552729481-4032800560-2279794651-1105 NTAP\group1 (Windows Domain group)
   S-1-5-21-3552729481-4032800560-2279794651-1107 NTAP\group3 (Windows Domain group)
   S-1-5-21-3552729481-4032800560-2279794651-111 NTAP\ProfGroup (Windows Domain group)
   S-1-1-8-2 Service asserted identity (Windows Well known group)
   S-1-5-32-551 BUILTIN\Backup Operators (Windows Alias)
   S-1-5-32-545 BUILTIN\Users (Windows Alias)

User is also a member of Everyone, Authenticated Users, and Network Users

Privileges (0x2086):
   SeBackupPrivilege
   SeRestorePrivilege
   SeChangeNotifyPrivilege
```
**LDAP Schema Templates**

The following shows the available schema templates in ONTAP. This output came from an ONTAP system that was running ONTAP 9.6.

### AD-IDMU

<table>
<thead>
<tr>
<th>Schema Template: AD-IDMU</th>
<th>Comment: Schema based on Active Directory Identity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management for UNIX (read-only)</td>
<td>RFC 2307 posixAccount Object Class: User</td>
</tr>
<tr>
<td>RFC 2307 posixGroup Object Class: Group</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 nisNetgroup Object Class: nisNetgroup</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 uid Attribute: uid</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 gidNumber Attribute: gidNumber</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 cn (for Groups) Attribute: cn</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 cn (for Netgroups) Attribute: name</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 userPassword Attribute: unixUserPassword</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 gecos Attribute: name</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 loginShell Attribute: loginShell</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 memberUid Attribute: memberUid</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 memberNisNetgroup Attribute: memberNisNetgroup</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 nisNetgroupTriple Attribute: nisNetgroupTriple</td>
<td></td>
</tr>
<tr>
<td>Enable Support for Draft RFC 2307bis: false</td>
<td></td>
</tr>
<tr>
<td>RFC 2307bis uniqueMember Attribute: uniqueMember</td>
<td></td>
</tr>
</tbody>
</table>

Data ONTAP Name Mapping windowsToUnix Object Class: User

Data ONTAP Name Mapping windowsAccount Attribute: msDS-PrincipalName

Data ONTAP Name Mapping windowsToUnix Attribute: sAMAccountName

No Domain Prefix for windowsToUnix Name Mapping: true

Vserver Owns Schema: true

Maximum groups supported when RFC 2307bis enabled: 256

### AD-SFU

<table>
<thead>
<tr>
<th>Schema Template: AD-SFU</th>
<th>Comment: Schema based on Active Directory Services for UNIX (read-only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2307 posixAccount Object Class: User</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 posixGroup Object Class: Group</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 nisNetgroup Object Class: msSFU30NisNetGroup</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 uid Attribute: sAMAccountName</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 gidNumber Attribute: msSFU30GidNumber</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 cn (for Groups) Attribute: cn</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 cn (for Netgroups) Attribute: name</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 userPassword Attribute: msSFU30Password</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 gecos Attribute: name</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 loginShell Attribute: msSFU30LoginShell</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 memberUid Attribute: msSFU30MemberUid</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 memberNisNetgroup Attribute: msSFU30MemberNisNetgroup</td>
<td></td>
</tr>
<tr>
<td>RFC 2307 nisNetgroupTriple Attribute: msSFU30MemberOfNisNetgroup</td>
<td></td>
</tr>
<tr>
<td>Enable Support for Draft RFC 2307bis: false</td>
<td></td>
</tr>
<tr>
<td>RFC 2307bis uniqueMember Attribute: uniqueMember</td>
<td></td>
</tr>
</tbody>
</table>

Data ONTAP Name Mapping windowsToUnix Object Class: User

Data ONTAP Name Mapping windowsAccount Attribute: windowsAccount

Data ONTAP Name Mapping windowsToUnix Attribute: windowsAccount

No Domain Prefix for windowsToUnix Name Mapping: false

Vserver Owns Schema: true

Maximum groups supported when RFC 2307bis enabled: 256
### RFC 2307 nisMapEntry Attribute: msSU30nisMapEntry

**MS-AD-BIS**

<table>
<thead>
<tr>
<th>Management for UNIX (read-only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2307 posixAccount Object Class: User</td>
</tr>
<tr>
<td>RFC 2307 posixGroup Object Class: Group</td>
</tr>
<tr>
<td>RFC 2307 nisNetgroup Object Class: nisNetgroup</td>
</tr>
<tr>
<td>RFC 2307 uid Attribute: uid</td>
</tr>
<tr>
<td>RFC 2307 uidNumber Attribute: uidNumber</td>
</tr>
<tr>
<td>RFC 2307 gidNumber Attribute: gidNumber</td>
</tr>
<tr>
<td>RFC 2307 cn (for Groups) Attribute: cn</td>
</tr>
<tr>
<td>RFC 2307 cn (for Netgroups) Attribute: name</td>
</tr>
<tr>
<td>RFC 2307 userPassword Attribute: unixUserPassword</td>
</tr>
<tr>
<td>RFC 2307 gecos Attribute: name</td>
</tr>
<tr>
<td>RFC 2307 loginShell Attribute: loginShell</td>
</tr>
<tr>
<td>RFC 2307 memberUid Attribute: memberUid</td>
</tr>
<tr>
<td>RFC 2307 memberNisNetgroup Attribute: memberNisNetgroup</td>
</tr>
<tr>
<td>RFC 2307 nisNetgroupTriple Attribute: nisNetgroupTriple</td>
</tr>
</tbody>
</table>

Enable Support for Draft RFC 2307bis: true

**RFC 2307bis groupOfUniqueNames Object Class: group**

**RFC 2307bis uniqueMember Attribute: Member**

**Data ONTAP Name Mapping windowsToUnix Object Class: User**

**Data ONTAP Name Mapping windowsAccount Attribute: sAMAccountName**

**No Domain Prefix for windowsToUnix Name Mapping: true**

### Maximum groups supported when RFC 2307bis enabled: 256

**RFC 2307 nisObject Object Class: nisObject**

**RFC 2307 nisMapName Attribute: nisMapName**

**RFC 2307 nisMapEntry Attribute: nisMapEntry**

### RFC 2307

<table>
<thead>
<tr>
<th>Comment: Schema based on RFC 2307 (read-only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2307 posixAccount Object Class: posixAccount</td>
</tr>
<tr>
<td>RFC 2307 posixGroup Object Class: posixGroup</td>
</tr>
<tr>
<td>RFC 2307 nisNetgroup Object Class: nisNetgroup</td>
</tr>
<tr>
<td>RFC 2307 uid Attribute: uid</td>
</tr>
<tr>
<td>RFC 2307 uidNumber Attribute: uidNumber</td>
</tr>
<tr>
<td>RFC 2307 gidNumber Attribute: gidNumber</td>
</tr>
<tr>
<td>RFC 2307 cn (for Groups) Attribute: cn</td>
</tr>
<tr>
<td>RFC 2307 cn (for Netgroups) Attribute: cn</td>
</tr>
<tr>
<td>RFC 2307 userPassword Attribute: userPassword</td>
</tr>
<tr>
<td>RFC 2307 gecos Attribute: gecos</td>
</tr>
<tr>
<td>RFC 2307 homeDirectory Attribute: homeDirectory</td>
</tr>
<tr>
<td>RFC 2307 loginShell Attribute: loginShell</td>
</tr>
<tr>
<td>RFC 2307 memberUid Attribute: memberUid</td>
</tr>
<tr>
<td>RFC 2307 memberNisNetgroup Attribute: memberNisNetgroup</td>
</tr>
<tr>
<td>RFC 2307 nisNetgroupTriple Attribute: nisNetgroupTriple</td>
</tr>
</tbody>
</table>

Enable Support for Draft RFC 2307bis: false

**RFC 2307bis groupOfUniqueNames Object Class: groupOfUniqueNames**

**RFC 2307bis uniqueMember Attribute: uniqueMember**

**Data ONTAP Name Mapping windowsToUnix Object Class: posixAccount**

**Data ONTAP Name Mapping windowsAccount Attribute: windowsAccount**

**No Domain Prefix for windowsToUnix Name Mapping: false**

**Vserver Owns Schema: true**

### Maximum groups supported when RFC 2307bis enabled: 256

**RFC 2307 nisObject Object Class: nisObject**

**RFC 2307 nisMapName Attribute: nisMapName**

**RFC 2307 nisMapEntry Attribute: nisMapEntry**
Sample LDAP Client Configuration

The following configuration is from an LDAP client pointing to Active Directory LDAP servers running Windows 2012.

Active Directory LDAP Client

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vserver:</td>
<td>DEMO</td>
</tr>
<tr>
<td>Client Configuration Name:</td>
<td>DEMO</td>
</tr>
<tr>
<td>LDAP Server List:</td>
<td>-</td>
</tr>
<tr>
<td>(DEPRECATED) LDAP Server List:</td>
<td>-</td>
</tr>
<tr>
<td>Preferred Active Directory Servers:</td>
<td>-</td>
</tr>
<tr>
<td>Active Directory Domain:</td>
<td>NTAP.LOCAL</td>
</tr>
<tr>
<td>Bind Using the Vserver's CIFS Credentials:</td>
<td>true</td>
</tr>
<tr>
<td>Schema Template:</td>
<td>DEMO</td>
</tr>
<tr>
<td>LDAP Server Port:</td>
<td>389</td>
</tr>
<tr>
<td>Query Timeout (sec):</td>
<td>3</td>
</tr>
<tr>
<td>Minimum Bind Authentication Level:</td>
<td>sasl</td>
</tr>
<tr>
<td>Bind DN (User):</td>
<td>ldap-user</td>
</tr>
<tr>
<td>Bind DN:</td>
<td></td>
</tr>
<tr>
<td>Base DN:</td>
<td></td>
</tr>
<tr>
<td>Base Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>User DN:</td>
<td>-</td>
</tr>
<tr>
<td>User Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Group DN:</td>
<td>-</td>
</tr>
<tr>
<td>Group Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Netgroup DN:</td>
<td>-</td>
</tr>
<tr>
<td>Netgroup Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Vserver Owns Configuration:</td>
<td>true</td>
</tr>
<tr>
<td>Use start-tls Over LDAP Connections:</td>
<td>false</td>
</tr>
<tr>
<td>Enable Netgroup-By-Host Lookup:</td>
<td>false</td>
</tr>
<tr>
<td>Netgroup-By-Host DN:</td>
<td>-</td>
</tr>
<tr>
<td>Netgroup-By-Host Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Client Session Security:</td>
<td>none</td>
</tr>
<tr>
<td>LDAP Referral Chasing:</td>
<td>false</td>
</tr>
<tr>
<td>Group Membership Filter:</td>
<td>-</td>
</tr>
</tbody>
</table>

The following configuration is from an LDAP client pointing to Red Hat Directory Server.

RHEL Directory Server LDAP Client

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client Configuration Name:</td>
<td>centos7.ntap2016.local</td>
</tr>
<tr>
<td>LDAP Server List:</td>
<td>centos7.ntap2016.local</td>
</tr>
<tr>
<td>(DEPRECATED) LDAP Server List:</td>
<td>-</td>
</tr>
<tr>
<td>Preferred Active Directory Servers:</td>
<td>-</td>
</tr>
<tr>
<td>Active Directory Domain:</td>
<td>-</td>
</tr>
<tr>
<td>Bind Using the Vserver's CIFS Credentials:</td>
<td>false</td>
</tr>
<tr>
<td>Schema Template:</td>
<td>RFC-2307</td>
</tr>
<tr>
<td>LDAP Server Port:</td>
<td>389</td>
</tr>
<tr>
<td>Query Timeout (sec):</td>
<td>3</td>
</tr>
<tr>
<td>Minimum Bind Authentication Level:</td>
<td>simple</td>
</tr>
<tr>
<td>Bind DN (User):</td>
<td>cn=ldapadm,dc=ntap2016,dc=local</td>
</tr>
<tr>
<td>Base DN:</td>
<td>dc=ntap2016,dc=local</td>
</tr>
<tr>
<td>Base Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>User DN:</td>
<td>-</td>
</tr>
<tr>
<td>User Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Group DN:</td>
<td>-</td>
</tr>
<tr>
<td>Group Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Netgroup DN:</td>
<td>-</td>
</tr>
<tr>
<td>Netgroup Search Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Vserver Owns Configuration:</td>
<td>false</td>
</tr>
<tr>
<td>Use start-tls Over LDAP Connections:</td>
<td>false</td>
</tr>
<tr>
<td>Enable Netgroup-By-Host Lookup:</td>
<td>false</td>
</tr>
<tr>
<td>Netgroup-By-Host DN:</td>
<td>-</td>
</tr>
<tr>
<td>Netgroup-By-Host Scope:</td>
<td>subtree</td>
</tr>
<tr>
<td>Client Session Security:</td>
<td>none</td>
</tr>
<tr>
<td>LDAP Referral Chasing:</td>
<td>false</td>
</tr>
<tr>
<td>Group Membership Filter:</td>
<td>-</td>
</tr>
</tbody>
</table>
LDAP Traffic as Seen from a Packet Trace

This section explains LDAP communication between an ONTAP SVM LDAP client and an LDAP server that runs Windows Active Directory. In this trace, `getxxbyyy` was run on the cluster for a specific user named `prof1`.

```plaintext
cluster::*> getxxbyyy -node node1 -vserv demo -username prof1 -show-source true -use-cache false
  (vserver services name-service getxxbyyy getpwbyname)
Source used for lookup: LDAP
pw_name: prof1
pw_passwd: *
pw_uid: 1100
pw_gid: 1101
pw_gecos:
pw_dir:
pw_shell: /bin/sh
```

Following is the LDAP client configuration:

```plaintext
cluster::*> ldap client show -client-config demo
  Vserver: demo
  Client Configuration Name: DEMO
  LDAP Server List: 
  (DEPRECATED) LDAP Server List: 
  Active Directory Domain: ntap.local
  Preferred Active Directory Servers: 
  Bind Using the Vserver's CIFS Credentials: true
  Schema Template: DEMO
  LDAP Server Port: 389
  Query Timeout (sec): 3
  Minimum Bind Authentication Level: sasl
    Bind DN (User): administrator
    Base DN: DC=NTAP,DC=local
    Base Search Scope: subtree
    User DN: CN=Users,DC=NTAP,DC=local
    User Search Scope: subtree
    Group DN: 
    Group Search Scope: subtree
    Netgroup DN: 
    Netgroup Search Scope: subtree
    Vserver Owns Configuration: true
    Use start-tls Over LDAP Connections: false
    Enable Netgroup-By-Host Lookup: true
    Netgroup-By-Host DN: 
    Netgroup-By-Host Scope: subtree
    Client Session Security: none
    LDAP Referral Chasing: false
    Group Membership Filter: 
```

The first steps in the trace were DNS record lookups for the LDAP SRV record. It is based on the `-ad-domain` setting that was used in the client configuration.

```
10.193.67.237 10.193.67.236 DNS 81 Standard query 0xcbec SRV _ldap._tcp.ntap.local
```

That request was successful, so another DNS request was made for the SRV record of the `Default-First-Site-Name`.

```
10.193.67.236 10.193.67.237 DNS 134 Standard query response 0xcbec SRV _ldap._tcp.ntap.local SRV 0 100 389 oneway.ntap.local A 10.193.67.236
10.193.67.237 10.193.67.236 DNS 112 Standard query 0x0075 SRV _ldap._tcp.Default-First-Site-Name._sites.ntap.local
10.193.67.236 10.193.67.237 DNS 165 Standard query response 0x0075 SRV _ldap._tcp.Default-First-Site-Name._sites.ntap.local SRV 0 100 389 oneway.ntap.local A 10.193.67.236
```
Next, the LDAP port that is specified in the client configuration is tested through a regular TCP packet to confirm that it is open and listening.

Next, a TCP packet is sent to open the port to perform the LDAP communication.

Then the LDAP bind occurs.

Because the LDAP client option is set as `-bind-as-cifs-server true`, the bind request uses the CIFS machine account and Kerberos authentication. You can see it in the packet details.

Kerberos is used here because the CIFS machine account named `DEMO` has a valid DNS A record and reverse name-lookup record. If for any reason you cannot use Kerberos, you fall back to NTLM.

The next packet shows that the bind is successful with Kerberos.
Now the LDAP search can be performed. Because it is a user request and \texttt{-user-dn} has been populated with a value, it is used for the search. If no user DN was provided, you fall back to the value that was set in \texttt{-base-dn}.

\begin{verbatim}
10.193.67.237 10.193.67.236 LDAP 239 searchRequest(2) "CN=Users,DC=NTAP,DC=local"

trace: 0000000007: request: searchRequest(2) "CN=Users,DC=NTAP,DC=local"

The LDAP \texttt{searchRequest} packet gives you a lot of information about the filters that were used in the search. If this LDAP request was using TLS or SSL, then it would be encrypted and you could not see the values unless you decrypted the trace.

In the following example, you can see that the user \texttt{objectClass} was used to search for the UID profl. You can also see that ONTAP asked for seven different attributes in the search. You can use this information to troubleshoot search requests and to build your own LDAP search filters for use with third-party tools such \texttt{ldapsearch} or Ldp.
The LDAP `searchRequest` replies with the requested information and shows that one result was returned.

```
10.193.67.236  10.193.67.237  LDAP  330  searchResEntry(2)
"CN=prof1,CN=Users,DC=NTAP,DC=local"  | searchResDone(2) success  [1 result]
```

In the trace details, you see the same output as from the `getXXbyXX` command from the ONTAP CLI.
In addition to viewing the LDAP search filters, you also can see how long the `searchRequest` took. Each packet is marked with a timestamp. In this case, you can see that the request returned in less than a second:

[Table showing packet details]

Having timestamp information helps you determine whether LDAP issues are occurring because of the configured LDAP timeout value in the LDAP client configuration.

Finally, when the request is finished, ONTAP closes the port request.

[Table showing closed port request details]

Contact Us

Let us know how we can improve this technical report. Contact us at docfeedback@netapp.com. Include TECHNICAL REPORT 4835 in the subject line.

Acknowledgments

Special thanks to Oliver Brakmann for providing a method to create bind-dn users in FreeIPA that are not the Directory Manager.

Where to Find Additional Information

- ONTAP 9 Documentation Center [https://docs.netapp.com/ontap-9/index.jsp](https://docs.netapp.com/ontap-9/index.jsp)

Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Document Version History</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td>May 2020</td>
<td>Initial version.</td>
</tr>
</tbody>
</table>
Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer’s installation in accordance with published specifications.

Copyright Information

Copyright © 2020 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

Data contained herein pertains to a commercial item (as defined in FAR 2.101) and is proprietary to NetApp, Inc. The U.S. Government has a non-exclusive, non-transferable, non-sublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b).

Trademark Information

NETAPP, the NETAPP logo, and the marks listed at http://www.netapp.com/TM are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.

TR-4835-0520