Red Hat Directory Server 12

Searching entries and tuning searches

Finding directory entries and improving search performance

Last Updated: 2024-01-17

Red Hat Directory Server 12 Searching entries and tuning searches

Finding directory entries and improving search performance

Legal Notice

Abstract

You can search for directory entries using the web console, command line, and by using the LDAP search utility. The search performance can be improved by using resource limits and can be set resource limits globally, at user level and for anonymous binds.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For submitting feedback through Jira (account required):
    1. Log in to the Jira website.
    2. Click **Create** in the top navigation bar
    3. Enter a descriptive title in the **Summary** field.
    4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
    5. Click **Create** at the bottom of the dialogue.
- For submitting feedback through Bugzilla (account required):
    1. Go to the Bugzilla website.
    2. As the Component, use **Documentation**.
    3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
    4. Click **Submit Bug**.

CHAPTER 1. FINDING ENTRIES USING THE COMMAND LINE (LDAPSEARCH)

You can use the **ldapsearch** command-line utility to search for directory entries. This utility opens a connection to a specified server using the specified identity and credentials and locates entries based on a specified search filter. The search scope can include:

- a single entry (**-s base**)
- an entry immediate subentries (**-s one**)
- an entire tree or subtree (**-s sub**)

NOTE

The **ldapsearch** utility does not search for directory entries based on attributes in the distinguished name. The distinguished name is only a unique identifier for a directory entry and cannot be used as a search key. Instead, **ldapsearch** searches for entries based on the attribute value pairs stored in entries. If the distinguished name of an entry is, for example, **uid=bjensen,ou=People,dc=example,dc=com**, then a search for **dc=example** does not match that entry unless **dc:example** was explicitly added as an attribute value pair to this entry.

The ldapsearch utility returns results in the LDIF format that is defined in the RFC 2849 specification.

1.1. THE LDAPSEARCH COMMAND FORMAT

The **ldapsearch** command must use the following format:

```
# ldapsearch [-x | -Y mechanism] [options] [search_filter] [list_of_attributes]
```

- **-x** or **-Y**
  Use **-x** (simple binds) or **-Y** (SASL mechanism) to configure the type of the connection.
- *options*
  The **ldapsearch** command-line options. Specify the options before the search filter, if any are used.
- *search_filter*
  An LDAP search filter. Do not specify a search filter if you configure search filters in a file using the **-f** option.
- *list_of_attributes*
  A list of attributes separated by a space character. Specifying the list of attributes reduces the number of attributes returned in the search results. This list of attributes must appear after the search filter. If you do not specify the list of attributes, the search returns values for all attributes permitted by the access control set in the directory with the exception of operational attributes.

  If you want the search to return operational attributes, you must explicitly specify it in the **ldapsearch** search command. To return all operational attributes of an object use **+**. To retrieve regular attributes in addition to explicitly specified operational attributes, use an asterisk (*) in the list of attributes.

  Note that you might need to escape the asterisk character with a backslash (*).

To retrieve only a list of matching DNs, use the attribute **1.1**. For example:

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com \
   -b "dc=example,dc=com" -x "(objectclass=inetorgperson)" 1.1
```

1.2. COMMONLY USED LDAPSEARCH OPTIONS

The following table lists the most commonly used **ldapsearch** utility options. If a specified value contains a space character, the value must be surrounded by single or double quotation marks, for example:

-b "cn=My Special Group,ou=groups,dc=example,dc=com"

IMPORTANT

The **ldapsearch** utility from OpenLDAP uses SASL connections by default. To perform a simple bind or to use TLS, use the **-x** argument to disable SASL and allow other connection methods.

Option	Description
-b	Specifies the starting point for the search - base Distinguished Name (DN). Note that distinguished name must exist in the database. If you set the **LDAP_BASEDN** environment variable as a base DN, you do not need to use this option. You must specify the option value in single or double quotation marks if the value contains a space character. For example: **-b "cn=user,ou=Product Development,dc=example,dc=com"**. To search the root DSE entry, specify an empty string here, such as **-b ""** .
-D	Specifies the DN used to authenticate to the server. Directory Server must recognize the DN value, and the DN must have the authority to search for the entries. For example: **-D "uid=user_name,dc=example,dc=com"**. Do not specify this option if the server supports anonymous access.

Option	Description

-H	Specifies an LDAP URL to connect to the server. An LDAP URL has the following format:
	
	`ldap[s]://hostname:[port]`
	
	Specifying the port value is optional. The **ldapsearch** utility will then use the default LDAP port 389 or LDAPS port 636.
	
	The utility can also use an LDAPI URL with each element separated by the HTML hex code %2F instead of a forward slash (/). For example:
	
	`ldapi://%2Ffull%2Fpath%2Fto%2Fslapd-example.socket`
	
	For LDAPI, specify the full path to the file which represents the LDAPI socket the server is listening to. If you did not specify the URL, **ldapsearch** uses the localhost or the setting specified in the **/etc/openldap/ldap.conf** file.

-h	Specifies the hostname or IP address of the machine with installed Directory Server. For example, **-h server.example.com**. If you did not specify a host, **ldapsearch** uses the localhost. Directory Server supports both IPv4 and IPv6 addresses. NOTE The **-h** option is deprecated and will be removed in a future release. Use the **-H** option instead.
-p	Specifies the TCP port number used by Directory Server. For example, **-p 1049**. The default port number is **389**. NOTE The **-p** option is deprecated and will be removed in a future release.
-l	Specifies the maximum time limit in seconds for a search request to complete. For example, **-l 300**. The time limit should not exceed the value specified in the **nsslapd-timelimit** attribute, because **ldapsearch** utility compares these two values and uses the smallest one. The default **nsslapd-timelimit** attribute value is **3600** seconds.

Option	Description
-s scope	Specifies the scope of the search. You can choose one of the following scopes: ● **sub** Searches through the entry specified in the **-b** option and all of its descendants entries. This is a default setting. ● **one** Searches through the immediate children of the entry specified in the **-b** option. The **ldapsearch** utility considers only children, not the base DN itself. ● **base** Searches only through the entry specified in the **-b** option or defined by the **LDAP_BASEDN** environment variable.
-W	Prompts for the password. if you did not specify the option, the **ldapsearch** utility uses anonymous access. Alternatively, use the **-w** option to pass the password to the utility. NOTE The password can be visible in the process list for other users and is saved in the shell's history.
-x	Disables the default SASL connection to allow simple binds.
-Y SASL_mechanism	Sets the SASL mechanism to use for authentication. If you do not set any mechanism, **ldapsearch** selects the best mechanism supported by the server. If you do not use the **-x** option, you must specify the**-Y** option instead.
-z number	Sets the maximum number of entries to return in a response to a search request. This value overwrites the **nsslapd-sizelimit** attribute when binding using the root DN.
-f	Specifies a file with search filters.

Additional resources

- nsslapd-timelimit description

- nsslapd-sizelimit description

1.3. USING SPECIAL CHARACTERS

When using the **ldapsearch** utility, you might need to specify values with characters that have special meaning to the command-line interpreter, such as space character, asterisk (*), or backslash (\). Depending on the command-line interpreter, enclose the value that has the special character either in single (**' '**) or double (**" "**) quotation marks. For example:

```
-D "cn=John Smith,ou=Product Development,dc=example,dc=com"
```

In general, use single quotation marks (**' '**) to enclose values. Use double quotation marks (**" "**) to allow variable interpolation if there are shell variables.

CHAPTER 2. FINDING ENTRIES USING THE WEB CONSOLE

You can search for directory entries using the web console.

2.1. FINDING ENTRIES USING THE LDAP BROWSER

You can use the LDAP Browser in the web console to search for entries in the Directory Server databases.

Directory Server searches for entries based on the attribute-value pairs stored in the entries, not based on the attributes used in the distinguished names (DN) of these entries. For example, if an entry has a DN of **uid=user_name,ou=People,dc=example,dc=com**, then a search for **dc=example** matches the entry only when **dc:example** attribute exists in this entry.

Prerequisites

- You are logged into the Directory Server web console.
- You have root permissions.

Procedure

1. In the web console, navigate to **LDAP Browser → Search**.
2. Expand and select the search criteria to filter entries:

Search parameter	Description
Search base	Specifies the starting point of the search. It is a distinguished name (DN) that currently exists in the database. **NOTE** The **Search** tabs opens with pre-defined search base, when you open an entry details in the **Tree View** or **Table View**, click on the Options menu (⋮) and select **Search**.
Search Scope	Select **Subtree** to search entries in the whole subtree starting from the search base and including all child entries. Select **One Level** to search entries starting from the search base and including only the first level of child entries. Select **Base** to search for attribute values only in the entry specified as the search base.

Search parameter	Description
Size Limit	Set the maximum number of entries to return from a search operation.
Time Limit	Set the time in seconds the search engine can look for entries.
Show Locking	Toggle the switch to **on** to see the lock status of the found entries.
Search Attributes	Select attributes that take part in the search. You can choose from the predefined attributes and add custom ones.

3. Type the attribute value in the search text field and press Enter.

4. Optional: To further refine your search, use search filters in the **Filter** tab to search for entries.

NOTE

Directory Server records all search requests to the access log file, which you can view at **Monitoring → Logging → Access Log**.

Additional resources

- nsslapd-timelimit description
- nsslapd-sizelimit description

CHAPTER 3. LDAP SEARCH FILTERS

Search filters select specific entries that search operation returns. You can use search filters with the **ldapsearch** command-line utility or in the Directory Server web console.

Directory Server searches for entries based on the attribute-value pairs the entries store, not based on the attributes used in the distinguished names (DN) of these entries. For example, if an entry has the DN **uid=user_name,ou=People,dc=example,dc=com**, then a search for **dc=example** matches the entry only when the attribute-value pair **dc:example** exists in this entry.

When using **ldapsearch**, you can define multiple search filters in a file with each filter on a separate line. Alternatively, you can specify a search filter directly on the command line.

A search filter has the following basic syntax:

```
<attribute><operator><value>
```

For example, the search filter **employeeNumber>=500** has **employeeNumber** as the attribute, **>=** as the operator, and **500** as the value.

You can define filters that use different attributes combined together with Boolean operators.

3.1. USING OPERATORS IN LDAP SEARCH FILTERS

Operators in LDAP search filters set the relationship between the attribute and the given search value. When searching for people, you can use operators to set a range, to return last names within a subset of letters in the alphabet or employee numbers that come after a certain number.

```
(employeeNumber>=500)
(sn~=suret)
(salary<=150000)
```

When having imperfect information or searching in internationalized directories, you can use operators for phonetic and approximate searches to make the search operation more effective.

You can use the following operators in the search filters:

Search type	Operator	Description
Equality	=	Returns entries with attributes which values exactly match the specified value. For example, **cn=example**.
Substring	=string* string	Returns entries that contain attributes with a specified substring in the value. For example, **cn=exa*l**. The asterisk (*) indicates zero (0) or more characters.

Search type	Operator	Description
Greater than or equal to	>=	Returns entries that contain attributes with values that are greater than or equal to the specified value. For example, **uidNumber>=5000**
Less than or equal to	⇐	Returns entries that contain attributes with values that are less than or equal to the specified value. For example, **uidNumber⇐5000**
Presence	=*	Returns entries that contain one or more values for the specified attribute. For example, **cn=***.
Approximate	~=	Returns entries that contain the specified attribute with a value that is approximately equal to the value specified in the search filter. For example, **l~=san fransico** returns **l=san francisco**.

3.2. USING COMPOUND LDAP SEARCH FILTERS

You can combine multiple LDAP search filter components by using Boolean operators expressed in the prefix notation as follows:

```
(<boolean-operator>(filter)(filter)(filter)...)
```

You can use the following Boolean operators:

Operator	Symbol	Description
AND	Ampersand (&)	All specified filters must be true for the statement to be true. For example, **(&(filter)(filter)(filter)…)**.
OR	Vertical bar (|)	At least one specified filter must be true for the statement to be true. For example, **(|(filter)(filter)(filter)…)**.

Operator	Symbol	Description
NOT	Exclamation point (!)	The specified statement must not be true for the statement to be true. Only one filter is affected by the NOT operator. For example, **(!(filter))**.

A search operation evaluates Boolean expressions in the following order:

- Innermost to outermost parenthetical expressions first
- All expressions from left to right

Compound search filters are most useful when they are nested together into completed expressions, such as:

```
(<boolean-operator>(filter)((<boolean-operator>(filter)(filter))))
```

You can combine compound filters with other types of searches (approximate, substring, and other operators) to get detailed results. The following example filter returns all entries which have the organizational unit (**ou**) as **Marketing** and which **description** attribute does not contain the substring **X.500**:

```
(&(ou=Marketing)(!(description=*X.500*)))
```

In addition, you can expand the filter to return also entries that have a **manager** set to **example** or **demo**:

```
(&(ou=Marketing)(!(description=*X.500*))(|
(manager=cn=example,ou=Marketing,dc=example,dc=com)
(manager=cn=demo,ou=Marketing,dc=example,dc=com)))
```

The following example filter returns all entries that do not represent a person:

```
(!(objectClass=person))
```

The following filter returns all entries that do not represent a person and which common name (**cn**) is similar to **printer3b**:

```
(&(!(objectClass=person))(cn~=printer3b))
```

CHAPTER 4. LDAP SEARCH (LDAPSEARCH) EXAMPLES

The following examples provide the most common `ldapsearch`es used for searching though the directory.

Prerequisites

- You perform the search for all entries in the directory.
- You configured the directory to support anonymous access for search and read operations. Therefore, you do not need to use **-W** and **-D** options in the command to supply any bind information. For more information on anonymous access, see Granting anonymous access.
- The server uses the default port number 389. You do not need to specify it in the search request.
- The server has the **server.example.com** hostname.
- You enabled TLS for the server on the port **636**, the default LDAPS port number.
- Directory Server store all data under the **dc=example,dc=com** suffix.

Returning all entries

The following LDAP search returns all entries in the directory:

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)"
```

Use the **(objectclass=*)** search filter to return every entry in the directory. Each entry must have an object class, and the **objectclass** attribute is always indexed.

Specifying search filters on the command line

You can specify a search filter directly on the command by enclosing the filter in quotation marks ("*filter*"). If you supply the filter in the command, do not specify the **-f** option. For example, to specify the **"cn=babs jensen"**, enter:

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "cn=babs
jensen"
```

Searching the Root DSE entry

The root DSE is a special entry that contains information about the directory server instance, including all of the suffixes that the local Directory Server supports. Search for this entry by supplying a search base of **""**, a search scope **base**, and the filter **"objectclass=*"**, for example:

```
# ldapsearch -H ldap://server.example.com -x -b "" -s base "objectclass=*"
```

Searching the schema entry

The **cn=schema** entry is a special entry that contains information about the directory schema, such as object classes and attribute types.

To list the content of the **cn=schema** entry, enter either of the following commands:

-

```
# ldapsearch -x -o ldif-wrap=no -b "cn=schema" \ '(objectClass=subSchema)' -s sub
objectClasses attributeTypes matchingRules \ matchingRuleUse dITStructureRules
nameForms ITContentRules ldapSyntaxes
```

or

```
# ldapsearch -x -o ldif-wrap=no -b "cn=schema" \ '(objectClass=subSchema)' -s sub "+"
```

Using LDAP_BASEDN variable

To simplify the search, you can set the search base by using the **LDAP_BASEDN** environment variable. You can set **LDAP_BASEDN** instead of using the **ldapsearch** command with the **-b** option. For more information about setting environment variables, see the documentation for the operating system.

Set **LDAP_BASEDN** to the directory suffix value. Because the directory suffix is equal to the root entry in the directory, all searches begin from the directory root entry.

For example, to set the **LDAP_BASEDN** variable to **dc=example,dc=com** and search for **cn=babs jensen** in the directory, enter:

```
# export LDAP_BASEDN="dc=example,dc=com"

# ldapsearch -H ldap://server.example.com -x "cn=babs jensen"
```

The command uses the default scope **sub** because the **-s** option was not supplied to specify the scope.

Displaying subsets of attributes

The **ldapsearch** command returns all search results in the LDIF format. By default, **ldapsearch** returns the entry distinguished name (DN) and all of the attributes that the user is allowed to read. You can set the directory access control to allow users to read only a subset of the attributes on any given directory entry.

Directory Server does not return operational attributes by default. To return operational attributes as a result of a search operation, explicitly specify these attributes in the search command or use the **+** argument to return all operational attributes. For more information, see Searching for operational attributes.

You can limit the returned attributes to a few specific attributes by specifying the required attributes on the command line after the search filter.

For example, to show the **cn** and **sn** attributes for every entry in the directory, enter:

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)" sn cn
```

Searching for operational attributes

Operational attributes are special attributes that Directory Server sets itself. Directory Server uses operational attributes to perform maintenance tasks, such as processing access control instructions. These attributes show specific information about the entry, such as the time this entry was initially created and the name of the user who created it.

You can use operational attributes on every entry in the directory, even if the attribute is specifically defined for the object class of the entry.

Regular **ldapsearch** commands do not return operational attributes. According to RFC3673, use **+** to return all operational attributes in a search request:

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)" '+'
```

To return only certain defined operational attributes, explicitly specify them in the **ldapsearch** request:

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)" creatorsName createTimestamp modifiersName modifyTimestamp
```

For the complete list of operational attributes, see Operational Attributes and Object Classes.

NOTE

To return all of the regular entry attributes along with the specified operational attributes, use the special search attribute, **"*"**, in addition to the operational attributes that you list.

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)" "*" aci
```

Note that you must enclose the asterisk (*) in quotation marks to prevent the shell from interpreting it.

Specifying search filters by using a file

You can specify search filters in a file instead of entering them on the command line.

Specify each search filter on a separate line in the file. The **ldapsearch** command runs each search in the order in which it appears in the file.

For example, the file contains the following filters:

```
sn=example
givenname=user
```

The **ldapsearch** command first finds all the entries with the **surname** set to **example**, then all the entries with the **givenname** set to **user**. If the search request finds an entry that matches both search criteria, then the entry is returned twice.

In the following search, the filters are specified in a file named **searchdb**:

```
# ldapsearch -H ldap://server.example.com -x -f searchdb
```

You can limit the set of returned attributes by specifying the attribute names at the end of the search line. For example, the following **ldapsearch** command performs both searches but returns only the DN and the **givenname** and **sn** attributes of each entry:

```
# ldapsearch -H ldap://server.example.com -x -f searchdb sn givenname
```

Specifying DNs that contain commas in search filters

When a DN within a search filter contains a comma as part of its value, the search command must escape the comma with a backslash (\). For example, to find everyone in the **example.com Bolivia, S.A.** subtree, enter:

```
# ldapsearch -H ldap://server.example.com -x -s base -b "l=Bolivia\, S.A.,dc=example,dc=com"
"objectclass=*"
```

Using the nsRole virtual attribute in the filter

In the following example, the **ldapsearch** command searches for DNs of all user entries that contain the **nsrole** attribute set to the **managed_role** value:

```
# ldapsearch -H ldap://server.example.com -x -b "dc=example,dc=com" "
(nsrole=cn=managed_role,dc=example,dc=com)" dn
```

Using a client certificate to bind to Directory Server

For more information about certificate-based authentication, see Configuring certificate-based authentication.

Searching with language matching rules

To explicitly submit a matching rule in a search filter, insert the matching rule after the attribute:

```
attr:matchingRule:=value
```

Matching rules are frequently used for searching internationalized directories. The following command searches for the department numbers after **N4709** in the Swedish (**2.16.840.1.113730.3.3.2.46.1**) matching rule.

```
departmentNumber:2.16.840.1.113730.3.3.2.46.1:=>= N4709
```

For more examples of performing internationalized searches, see Searching an Internationalized Directory.

Searching for attributes with bit field values

Bitwise searches use the bitwise **AND** or bitwise **OR** matching rules to perform bitwise search operations on attributes with values that are bit fields.

NOTE

Attributes with values for bit fields are not common in LDAP. Default Directory Server schemas do not use bit fields as attribute syntax. However, several LDAP syntaxes support integer-style values. You can define custom attributes to use bit field values. Applications can use custom attributes to perform bitwise operations against bit field values.

The bitwise **AND** matching rule (**1.2.840.113556.1.4.803**) checks if the bit given in the assertion value is set in the bit field attribute value. It is similar to an equality search. The following example sets **userAccountControl** value to the bit that represents **2**:

```
"(UserAccountControl:1.2.840.113556.1.4.803:=2)"
```

The following example show that the **userAccountControl** value must have all of the bits that are set in the value **6** (bits **2** and **4**):

```
"(UserAccountControl:1.2.840.113556.1.4.803:=6)”
```

The bitwise **OR** matching rule (**1.2.840.113556.1.4.804**) checks if any of the bits in the assertion string are represented in the attribute value. It is similar to a substring search. In this example, the **UserAccountControl** value must have any of the bits that are set in the bit field of **6**, meaning that the attribute value can be **2**, **4**, or **6**:

```
"(UserAccountControl:1.2.840.113556.1.4.804:=6)"
```

You can use bitwise searches with the Windows-Linux integration, such as using Samba file servers.

Additional resources

- The **ldapsearch** command format
- Commonly used **ldapsearch** options

CHAPTER 5. IMPROVING SEARCH PERFORMANCE THROUGH RESOURCE LIMITS

Searching through every entry in a database can have a negative impact on a server performance for larger directories. In large databases, effective indexing might not sufficiently reduce the search scope to improve the performance.

You can set limits on user and client accounts to reduce the total number of entries or the total amount of time spent in an individual search. This makes searches more responsive and improves overall server performance.

5.1. SEARCH OPERATION LIMITS FOR LARGE DIRECTORIES

You can control server limits for search operations by using special operational attribute values on the client application binding to the directory. You can set the following search operation limits:

- The **Look through** limit specifies how many entries you can examine for a search operation.
- The **Size** limit specifies maximum number of entries the server returns to a client application in response to the search operation.
- The **Time** limit specifies maximum time the server can spend processing a search operation.
- The **Idle timeout** limit specifies the time when connection to the server can be idle before the connection is dropped.
- The **Range timeout** limit specifies a separate **look-through** limit specifically for searches by using a range.

In the global server configuration, the resource limits set for the client application take precedence over the default resource limits set.

NOTE

The Directory Manager receives unlimited resources by default, with the exception of range searches.

5.2. SEARCH PERFORMANCE IMPROVEMENT WITH INDEX SCAN LIMITS

For large indexes, it is efficient to treat any search which matches the index as an unindexed search. The search operation has to look in the entire directory to process results rather than searching through an index that is nearly the size of a directory in addition to the directory itself.

Additional resources

- Setting an index scan limit

5.3. FINE GRAINED ID LIST SIZE

In large databases, some queries can consume a large number of CPU and RAM resources. To improve the performance, you can set a default ID scan limit that applies to all indexes in the database by using the **nsslapd-idlistscanlimit** attribute. However, it is useful to either define a limit for certain indexes or use the list with no IDs defined. You can set individual settings for ID list scan limits for different types of search filters by using the **nsIndexIDListScanLimit** attribute.

Additional resources

- Setting an index scan limit to improve performance when loading long list of ids

5.4. SETTING USER AND GLOBAL RESOURCE LIMITS BY USING THE COMMAND LINE

You can set **user-level** resource limits, **global resource** limits, and limits for specific types of searches, such as **simple paged** and **range searches**, by using the command line. You can set user-level attributes on the individual entries and global configuration attributes are set in the appropriate server configuration area.

You can set the following mentioned operational attributes for each entry by using the **ldapmodify** command:

- **look-through**
  You can specify the number of entries to examine for a search operation by using the **look-through** limit attribute. Setting the attribute's value to **-1** indicates that there is no limit.

  1. User-level attribute: **nsLookThroughLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-lookthroughlimit**
     b. Entry: **cn=config,cn=ldbm database,cn=plugins,cn=config**

     ```
     # dsconf instance backend config set --lookthroughlimit value
     ```

- **paged look-through**
  You can specify the number of entries to examine for simple paged search operations by using the **paged look-through** limit attribute. Setting the attribute's value to **-1** indicates that there is no limit.

  1. User-level attribute: **nsPagedLookThroughLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-pagedlookthroughlimit**
     b. Entry: **cn=config,cn=ldbm database,cn=plugins,cn=config**

     ```
     # dsconf instance backend config set --pagedlookthroughlimit value
     ```

- **size**
  You can specify the maximum number of entries the server returns to a client application in response to a search operation by using the **size** limit attribute. Setting the attribute's value to **-1** indicates that there is no limit.

  1. User-level attribute: **nsSizeLimit**
  2. Global configuration:

   a. Attribute: **nsslapd-sizelimit**
   b. Entry: **cn=config**

   ```
   # dsconf instance config replace nsslapd-sizelimit value
   ```

   You can add the **nsSizeLimit** attribute to the user's entry and for example give it a search return size limit of **500** entries:

   ```
   # ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
   ...
   dn: uid=user_name,ou=People,dc=example,dc=com
   changetype: modify
   add: nsSizeLimit
   nsSizeLimit: 500
   ...
   ```

- **paged size**
  You can specify the maximum number of entries the server returns to a client application for simple paged search operations by using the **paged size** limit attribute. Setting the attribute's value to **-1** indicates that there is no limit.

  1. User-level attribute: **nsPagedSizeLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-pagedsizelimit**
     b. Entry: **cn=config**

     ```
     # dsconf instance config replace nsslapd-pagedsizelimit value
     ```

- **time**
  You can specify the maximum time the server can spend processing a search operation by using the **time** limit attribute. Setting the attribute's value to **-1** indicates that there is no time limit.

  1. User-level attribute: **nsTimeLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-timelimit**
     b. Entry: **cn=config**

     ```
     # dsconf instance config replace nsslapd-timelimit value
     ```

- **idle timeout**
  You can specify the time in seconds for which a connection to the server can be idle before the connection is dropped by using the **idle timeout** attribute. Setting the attribute's value to **-1** indicates that there is no limit.

  1. User-level attribute: **nsidletimeout**
  2. Global configuration:

a. Attribute: **nsslapd-idletimeout**

b. Entry: **cn=config**

```
# dsconf instance config replace nsslapd-idletimeout value
```

- **ID list scan**
  You can specify the maximum number of entry IDs loaded from an index file for search results. If the ID list size is greater than the maximum number of IDs, the search will not use the index list, but will treat the search as an unindexed search and look through the entire database.

  1. User-level attribute: **nsIDListScanLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-idlistscanlimit**
     b. Entry: **cn=config,cn=ldbm database,cn=plugins,cn=config**

```
# dsconf instance backend config set --idlistscanlimit value
```

- **paged ID list scan**
  You can specify the maximum number of entry IDs loaded from an index file for search results particularly for paged search operations by using the **paged ID list scan** limit.

  1. User-level attribute: **nsPagedIDListScanLimit**
  2. Global configuration:
     a. Attribute: **nsslapd-pagedidlistscanlimit**
     b. Entry: **cn=config,cn=ldbm database,cn=plugins,cn=config**

```
# dsconf instance backend config set --pagedidlistscanlimit value
```

- **range look-through**
  You can specify the numbers of entries to examine for a range search operation by using the **range look-through** limit. Setting the attribute's value to **-1** indicates that there is no limit.

  NOTE

  A range search is a search by using the **greater-than**, **equal-to-or-greater-than**, **less-than**, or **equal-to-less-than** operators.

  1. User-level attribute: **not available**
  2. Global configuration:
     a. Attribute: **nsslapd-rangelookthroughlimit**
     b. Entry: **cn=config,cn=ldbm database,cn=plugins,cn=config**

```
# dsconf instance backend config set ----rangelookthroughlimit value
```

NOTE

You can set an access control list to prevent users from changing the setting.

Additional resources

- Managing access control

5.5. SETTING RESOURCE LIMITS ON ANONYMOUS BINDS

You can configure resource limits for anonymous binds by creating a template user entry that has resource limits, and then applying this template to anonymous binds, because resource limits are set on a user entry and anonymous bind does not have a user entry associated with it.

Prerequisites

- A template entry has been created.

Procedure

1. Set resource limits you want to apply to anonymous binds:

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
...
dn: cn=anonymous_template,ou=people,dc=example,dc=com
objectclass: nsContainer
objectclass: top
cn: anonymous_template
nsSizeLimit: 250
nsLookThroughLimit: 1000
nsTimeLimit: 60
...
```

NOTE

For performance reasons, the template must be in the normal back end, not in the **cn=config** suffix that does not use an entry cache.

2. Add the **nsslapd-anonlimitsdn** parameter to the server configuration, pointing to the **DN** of the template entry on all suppliers in a replication topology:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-anonlimitsdn="cn=anonymous_template,ou=people,dc=example,dc=com"
```

5.6. PERFORMANCE IMPROVEMENT FOR RANGE SEARCHES

A range search (all IDs search) uses operators to set a bracket to search and return an entire subset of the entries within a directory. The range search can evaluate every entry in the directory to check if the entry is within the provided range.

For example, to search for every entry modified at or after midnight on January 1, run the following command:

```
# (modifyTimestamp>=20210101010101Z)
```

To prevent a range search from turning into an all IDs search, you can use the **look-through** limit. By using this limit, you can improve overall performance and speed up range search results. However, some clients or administrative users, such as Directory Manager, cannot have the **look-through** limit set. In this case, the range search can take several minutes to complete or can even continue indefinitely.

However, you can set a separate range **look-through** limit. By setting this limit, clients and administrative users can have high **look-through** limits and can still be able to set a reasonable limit on potentially performance-impaired range searches.

You can configure such setting by using the **nsslapd-rangelookthroughlimit** attribute. The default value is 5000.

To set the separate range **look-through** limit to 7500, run the following command:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
rangelookthroughlimit 7500
```