Red Hat Directory Server 12

Managing directory attributes and values

Managing directory entries using ldapadd, ldapmodify, ldapdelete, and dsconf utilities or web console

Last Updated: 2024-10-11

Red Hat Directory Server 12 Managing directory attributes and values

Managing directory entries using ldapadd, ldapmodify, ldapdelete, and dsconf utilities or web console

Legal Notice

Abstract

Learn how to manage Directory Server entries by using tools from the openldap-clients package. Enforce attribute uniqueness, assign class of services (CoS) to simplify entry management, reduce storage requirements, and avoid replication conflicts.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DIRECTORY SERVER

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

- For submitting feedback on the Red Hat Directory Server documentation through Jira (account required):
  1. Go to the Red Hat Issue Tracker.
  2. Enter a descriptive title in the **Summary** field.
  3. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
  4. Click **Create** at the bottom of the dialogue.
- For submitting feedback on the Red Hat Directory Server product through Jira (account required):
  1. Go to the Red Hat Issue Tracker.
  2. On the **Create Issue** page, click **Next**.
  3. Fill in the **Summary** field.
  4. Select the component in the **Component** field.
  5. Fill in the **Description** field including:
     a. The version number of the selected component.
     b. Steps to reproduce the problem or your suggestion for improvement.
  6. Click **Create**.

CHAPTER 1. MANAGING DIRECTORY ENTRIES USING THE COMMAND LINE

You can add, edit, rename, and delete an LDAP entry using the command line.

1.1. PROVIDING INPUT TO THE LDAPADD, LDAPMODIFY, AND LDAPDELETE UTILITIES

When you add, update, or delete entries or attributes in the directory, you can either use the interactive mode of the utilities to enter LDAP Data Interchange Format (LDIF) statements or pass an LDIF file to them.

1.1.1. The interactive mode of OpenLDAP client utilities

In the interactive mode, the **ldapadd**, **ldapmodify**, and **ldapdelete** utilities read the input from the command line. To exit the interactive mode, press the **Ctrl**+**D** (^D) key combination to send the end-of-file (EOF) escape sequence.

In interactive mode, the utility sends the statements to the LDAP server when you press **Enter** twice or when you send the EOF sequence.

Use the interactive mode:

- To enter LDAP Data Interchange Format (LDIF) statements without creating a file.

  Example 1.1. Using the ldapmodify interactive mode to enter LDIF statements

  The following example runs **ldapmodify** in interactive mode, deletes the **telephoneNumber** attribute, and adds the **manager** attribute with the **cn=manager_name,ou=people,dc=example,dc=com** value to the **uid=user,ou=people,dc=example,dc=com** entry. Press **Ctrl+D** after the last statement to exit the interactive mode.

  ```
  # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

  dn: uid=user,ou=people,dc=example,dc=com
  changetype: modify
  delete: telephoneNumber
  -
  add: manager
  manager: cn=manager_name,ou=people,dc=example,dc=com

  modifying entry "uid=user,ou=people,dc=example,dc=com"

  ^D
  ```

- To redirect LDIF statements, outputted by an another command, to the server:

  Example 1.2. Using the ldapmodify interactive mode with redirected content

  The following example redirects the output of the **command_that_outputs_LDIF** command to **ldapmodify**. The interactive mode exits automatically after the redirected command exits.

```
# command_that_outputs_LDIF | ldapmodify -D "cn=Directory Manager" -W -H
ldap://server.example.com -x
```

Additional resources

- **ldif(5)** man page

1.1.2. The file mode of OpenLDAP client utilities

In the interactive mode, the **ldapadd**, **ldapmodify**, and **ldapdelete** utilities read the LDAP Data Interchange Format (LDIF) statements from a file. Use this mode to send a larger number of LDIF statements to the server.

Example 1.3. Passing a File with LDIF Statements to ldapmodify

1. Create a file with the LDIF statements. For example, create the **~/example.ldif** file with the following statements:

```
dn: uid=user,ou=people,dc=example,dc=com
changetype: modify
delete: telephoneNumber
-
add: manager
manager: cn=manager_name,ou=people,dc=example,dc=com
```

   This example deletes the **telephoneNumber** attribute and to adds the **manager** attribute with the **cn=manager_name,ou=people,dc=example,dc=com** value to the **uid=user,ou=people,dc=example,dc=com** entry.

2. Pass the file to the **ldapmodify** command using the **-f** parameter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x -f
~/example.ldif
```

Additional resources

- **ldif(5)** man page

1.1.3. The continuous operation mode of OpenLDAP client utilities

By default, if you send multiple LDAP Data Interchange Format (LDIF) statements to the server and one operation fails, the process stops. However, entries processed before the error occurred were successfully added, modified, or deleted.

To ignore errors and continue processing further LDIF statements in a batch, pass the **-c** parameter to **ldapadd** and **ldapmodify**:

```
# ldpamodify -c -D "cn=Directory Manager" -W -H ldap://server.example.com -x
```

1.2. ADDING AN LDAP ENTRY USING THE COMMAND LINE

To add a new entry to the directory, use the **ldapadd** or **ldapmodify** utility. Note that **/bin/ldapadd** is a symbolic link to **/bin/ldapmodify**. Therefore, **ldapadd** performs the same operation as **ldapmodify -a**.

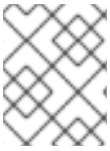
NOTE

You can only add a new directory entry if the parent entry already exists. For example, you cannot add **cn=user,ou=people,dc=example,dc=com**, if the **ou=people,dc=example,dc=com** parent entry does not exist.

1.2.1. Adding an entry using ldapadd

To use the **ldapadd** utility to add, for example, the **cn=user,ou=people,dc=example,dc=com** user entry, enter:

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
uid: user
givenName: given_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: surname
cn: user
```



NOTE

Running **ldapadd** automatically performs a **changetype: add** operation. Therefore, you do not need to specify **changetype: add** in the LDIF statement.

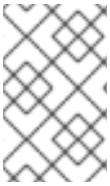
Additional resources

- **ldapadd(1)** man page

1.2.2. Adding an entry using ldapmodify

To use the **ldapmodify** utility to add, for example, the **cn=user,ou=people,dc=example,dc=com** user entry, enter:

```
# ldapmodify -a -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
uid: user
givenName: given_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: surname
cn: user
```



NOTE

When passing the **-a** parameter to the **ldapmodify** command, the utility automatically performs a **changetype: add** operation. Therefore, you do not need to specify **changetype: add** in the LDIF statement.

Additional resources

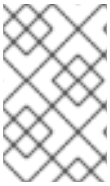
- **ldapmodify(1)** man page

1.2.3. Creating a root entry of a database suffix

To create the root entry of a database suffix, such as **dc=example,dc=com**, bind as the **cn=Directory Manager** user and add the entry. The distinguished name (DN) corresponds to the DN of the root or sub-suffix of the database.

For example, to add the **dc=example,dc=com** suffix, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: add
objectClass: top
objectClass: domain
dc: example
```



NOTE

You can add root objects only if you have one database per suffix. If you create a suffix that is stored in several databases, you must use the **dsctl ldif2db** command to set the database that will hold the new entries.

Additional resources

- Importing data using the command line while the server is offline

1.3. UPDATING AN LDAP ENTRY USING THE COMMAND LINE

When you modify a directory entry, use the **changetype: modify** statement. Depending on the change operation, you can add, change, or delete attributes from the entry.

1.3.1. Adding attributes to an LDAP entry

To add an attribute to an LDAP entry, use the **add** operation.

For example, to add the **telephoneNumber** attribute with the **555-1234567** value to the **uid=user,ou=People,dc=example,dc=com** entry, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
```

```
changetype: modify
add: telephoneNumber
telephoneNumber: 555-1234567
```

If an attribute is multi-valued, you can specify the attribute name multiple times to add all the values in a single operation. For example, to add two **telephoneNumber** attributes at once to the **uid=user,ou=People,dc=example,dc=com**, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
add: telephoneNumber
telephoneNumber: 555-1234567
telephoneNumber: 555-7654321
```

1.3.2. Updating the value of an attribute

The procedure for updating an attribute's value depends on whether the attribute is single-valued or multi-valued:

- Updating a single-value attribute:
  When updating a single-value attribute, use the **replace** operation to override the existing value. The following command updates the **manager** attribute of the **uid=user,ou=People,dc=example,dc=com** entry:

  ```
  # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

  dn: uid=user,ou=People,dc=example,dc=com
  changetype: modify
  replace: manager
  manager: uid=manager_name,ou=People,dc=example,dc=com
  ```

- Updating a specific value of a multi-value attribute:
  To update a specific value of a multi-value attribute, first delete the entry you want to replace, and then add the new value. The following command updates only the **telephoneNumber** attribute that is currently set to **555-1234567** in the **uid=user,ou=People,dc=example,dc=com** entry:

  ```
  # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

  dn: uid=user,ou=People,dc=example,dc=com
  changetype: modify
  delete: telephoneNumber
  telephoneNumber: 555-1234567
  -
  add: telephoneNumber
  telephoneNumber: 555-9876543
  ```

1.3.3. Deleting attributes from an entry

To delete an attribute from an entry, use the **delete** operation:

- Deleting an attribute:
  For example, to delete the **manager** attribute from the **uid=user,ou=People,dc=example,dc=com** entry, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
delete: manager
```

IMPORTANT

If the attribute contains multiple values, this operation deletes all of them.

- Deleting a specific value of a multi-value attribute:
  If you want to delete a specific value from a multi-value attribute, list the attribute and its value in the LDAP Data Interchange Format (LDIF) statement. For example, to delete only the **telephoneNumber** attribute that is set to **555-1234567** from the **uid=user,ou=People,dc=example,dc=com** entry, enter:

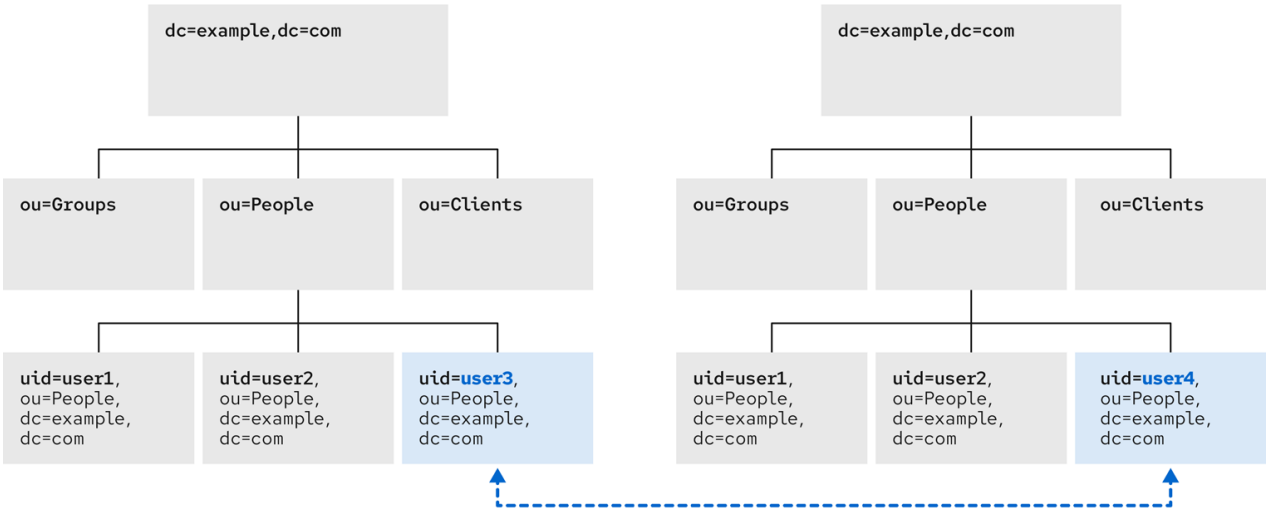
```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
delete: telephoneNumber
telephoneNumber: 555-1234567
```

1.4. RENAMING AND MOVING AN LDAP ENTRY

The following rename operations exist:

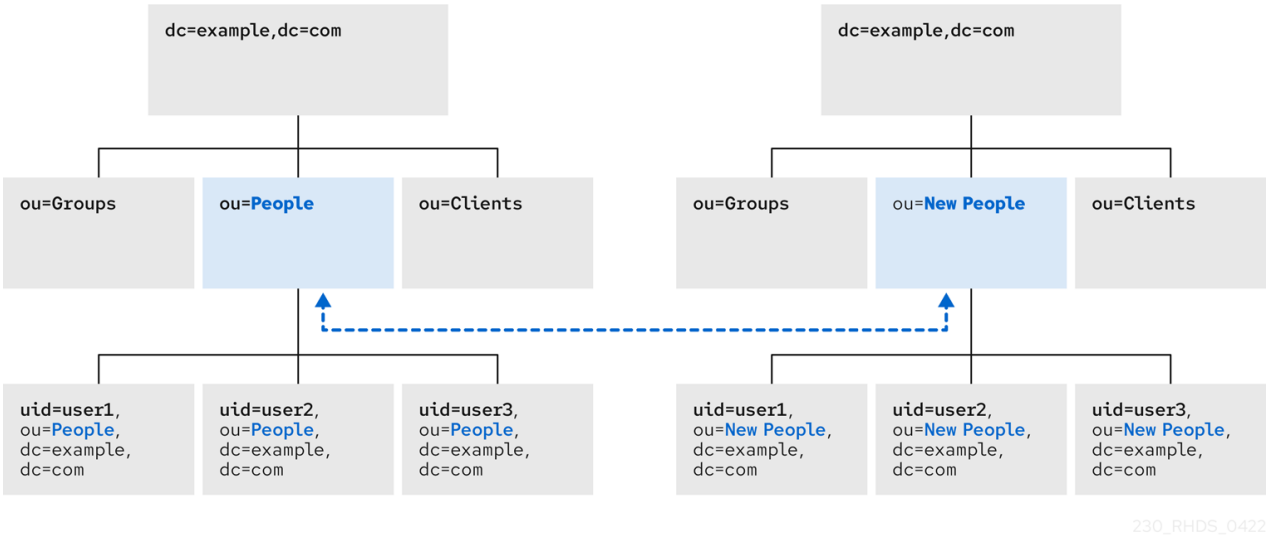
Renaming an entry

If you rename an entry, the **modrdn** operation changes the relative distinguished name (RDN) of the entry:



Renaming a subentry

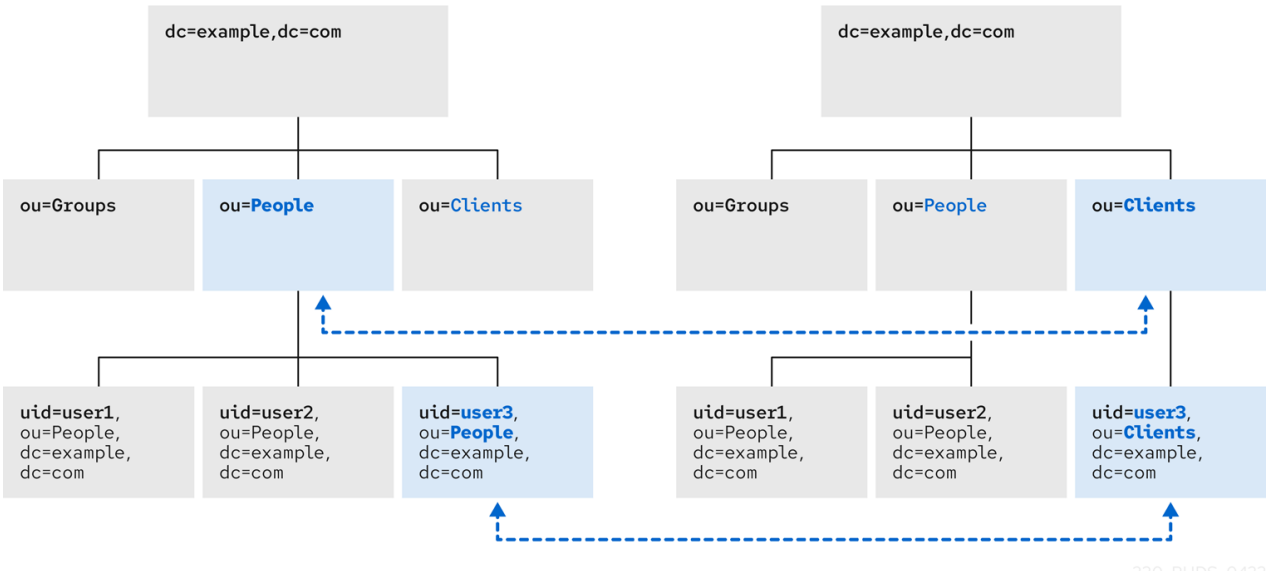
For subtree entries, the **modrdn** operation renames the subtree and also the DN components of child entries:



Note that for large subtrees, this process can take a lot of time and resources.

Moving an entry to a new parent

A similar action to renaming a subtree is moving an entry from one subtree to another. This is an expanded type of the **modrdn** operation, which simultaneously renames the entry and sets a **newSuperior** attribute which moves the entry from one parent to another:



1.4.1. Considerations for renaming LDAP entries

Keep the following in mind when performing rename operations:

- You cannot rename the root suffix.
- Subtree rename operations have minimal effect on replication. Replication agreements are applied to an entire database, not to a subtree within the database. Therefore, a subtree rename

operation does not require re-configuring a replication agreement. All name changes after a subtree rename operation are replicated as normal.

- Renaming a subtree might require any synchronization agreements to be reconfigured. Synchronization agreements are set at the suffix or subtree level. Therefore, renaming a subtree can break synchronization.

- Renaming a subtree requires that any subtree-level access control instructions (ACI) set for the subtree be reconfigured manually, as well as any entry-level ACIs set for child entries of the subtree.

- Trying to change the component of a subtree, such as moving from **ou** to **dc**, might fail with a schema violation. For example, the **organizationalUnit** object class requires the **ou** attribute. If that attribute is removed as part of renaming the subtree, the operation fails.

- If you move a group, the **MemberOf** plug-in automatically updates the **memberOf** attributes. However, if you move a subtree that contain groups, you must manually create a task in the **cn=memberof** task entry or use the **dsconf memberof fixup** command to update the related **memberOf** attributes.

1.4.2. Controlling the relative distinguished name behavior when renaming entries

When you rename an entry, the **deleteOldRDN** attribute controls whether the old relative distinguished name (RDN) will be deleted or retained:

deleteOldRDN: 0

The existing RDN is retained as a value in the new entry. The resulting entry contains two **cn** attributes: one with the old and one with the new common name (CN).
For example, the following attributes belong to a group that was renamed from **cn=old_group,dc=example,dc=com** to **cn=new_group,dc=example,dc=com** with the **deleteOldRDN** attribute set to **0**:

```
dn: cn=new_group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
cn: old_group
cn: new_group
```

deleteOldRDN: 1

Directory Server deletes the old entry and creates a new entry using the new RDN. The new entry only contains the **cn** attribute of the new entry.
For example, the following group was renamed to **cn=new_group,dc=example,dc=com** with the **deleteOldRDN** attribute set to **1**:

```
dn: cn=new_group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupofuniquenames
cn: new_group
```

Additional resources

- Renaming an LDAP entry or subtree

1.4.3. Renaming an LDAP entry or subtree

To rename an entry or subtree, use the **changetype: modrdn** operation, and set the new relative distinguished name (RDN) in the **newrdn** attribute.

For example, to rename the **cn=demo1,dc=example,dc=com** entry to **cn=demo2,dc=example,dc=com**, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=demo1,dc=example,dc=com
changetype: modrdn
newrdn: cn=demo2
deleteOldRDN: 1
```

Additional resources

- Controlling the relative distinguished name behavior when renaming entries

1.4.4. Moving an LDAP entry to a new parent

To move an entry to a new parent, use the **changetype: modrdn** operation, and set the following to attributes:

- **newrdn**: Sets the relative distinguished name (RDN) of the moved entry. You must set this entry, even if the RDN remains the same.
- **newSuperior**: Sets the distinguished name (DN) of the new parent entry.

For example, to move the **cn=demo** entry from **ou=Germany,dc=example,dc=com** to **ou=France,dc=example,dc=com**, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=demo,ou=Germany,dc=example,dc=com
changetype: modrdn
newrdn: cn=demo
newSuperior: ou=France,dc=example,dc=com
deleteOldRDN: 1
```

Additional resources

- Controlling the relative distinguished name behavior when renaming entries

1.5. DELETING AN LDAP ENTRY USING THE COMMAND LINE

You can remove entries from an LDAP directory, but you can only delete entries that have no child entries. For example, you cannot delete **ou=People,dc=example,dc=com**, if the **uid=user,ou=People,dc=example,dc=com** entry still exists.

1.5.1. Deleting an entry using ldapdelete

The **ldapdelete** utility enables you to delete one or multiple entries. For example, to delete the **uid=user,ou=People,dc=example,dc=com** entry, enter:

```
# ldapdelete -D "cn=Directory Manager" -W -H ldap://server.example.com -x
"uid=user,ou=People,dc=example,dc=com"
```

To delete multiple entries in one operation, append them to the command:

```
# ldapdelete -D "cn=Directory Manager" -W -H ldap://server.example.com -x
"uid=user1,ou=People,dc=example,dc=com" "uid=user2,ou=People,dc=example,dc=com"
```

Additional resources

- **ldapdelete(1)** man page

1.5.2. Deleting an entry using ldapmodify

To delete an entry using the **ldapmodify** utility, use the **changetype: delete** operation. For example, to delete the **uid=user,ou=People,dc=example,dc=com** entry, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: delete
```

1.6. USING SPECIAL CHARACTERS IN OPENLDAP CLIENT UTILITIES

When using the command line, enclose characters that have a special meaning to the command-line interpreter, such as space ( ), asterisk (*), or backslash (\), with quotation marks. Depending on the command-line interpreter, use single or double quotation marks. For example, to authenticate as the **cn=Directory Manager** user, enclose the user's distinguished name (DN) in quotation marks:

```
# ldapmodify -a -D "cn=Directory Manager" -W -H ldap://server.example.com -x
```

Additionally, if a DN contains a comma in a component, escape it using a backslash. For example, to authenticate as the **uid=user,ou=People,dc=example.com Chicago, IL** user, enter:

```
# ldapmodify -a -D "cn=uid=user,ou=People,dc=example.com Chicago\, IL" -W -H
ldap://server.example.com -x
```

1.7. USING BINARY ATTRIBUTES IN LDIF STATEMENTS

Certain attributes support binary values, such as the **jpegPhoto** attribute. When you add or update such an attribute, the utility reads the value for the attribute from a file. To add or update such an attribute, you can use the **ldapmodify** utility.

For example, to add the **jpegPhoto** attribute to the **uid=user,ou=People,dc=example,dc=com** entry, and read the value for the attribute from the **/home/user_name/photo.jpg** file, enter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
```

```
changetype: modify
add: jpegPhoto
jpegPhoto:< file:///home/user_name/photo.jpg
```

IMPORTANT

Note that there is no space between **:** and **<**.

1.8. UPDATING AN LDAP ENTRY IN AN INTERNATIONALIZED DIRECTORY

To use attribute values with languages other than English, associate the attribute's value with a language tag.

When using **ldapmodify** to update an attribute that has a language tag set, you must match the value and language tag exactly or the operation will fail.

For example, to modify an attribute value that has the **lang-fr** language tag set, include the tag in the modify operation:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
replace: homePostalAddress;lang-fr
homePostalAddress;lang-fr: 34 rue de Seine
```

CHAPTER 2. MANAGING DIRECTORY ENTRIES USING THE WEB CONSOLE

You can add, edit, rename, and delete LDAP entries using the web console.

2.1. ADDING AN LDAP ENTRY USING THE WEB CONSOLE

You can create the following entries using the web console:

- users
- groups
- roles
- organizational units (OUs)
- custom entries

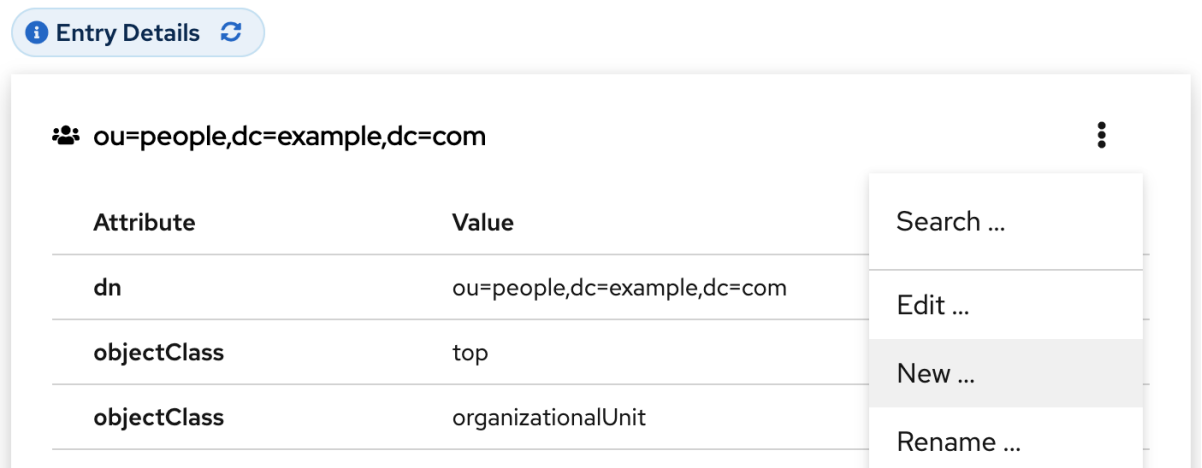
For example, you want to create a POSIX user **cn=John Smith,ou=people,dc=example,dc=com** with a password.

Prerequisites

- You are logged into the Directory Server web console.
- The parent entry exists. For example, **ou=people,dc=example,dc=com**.

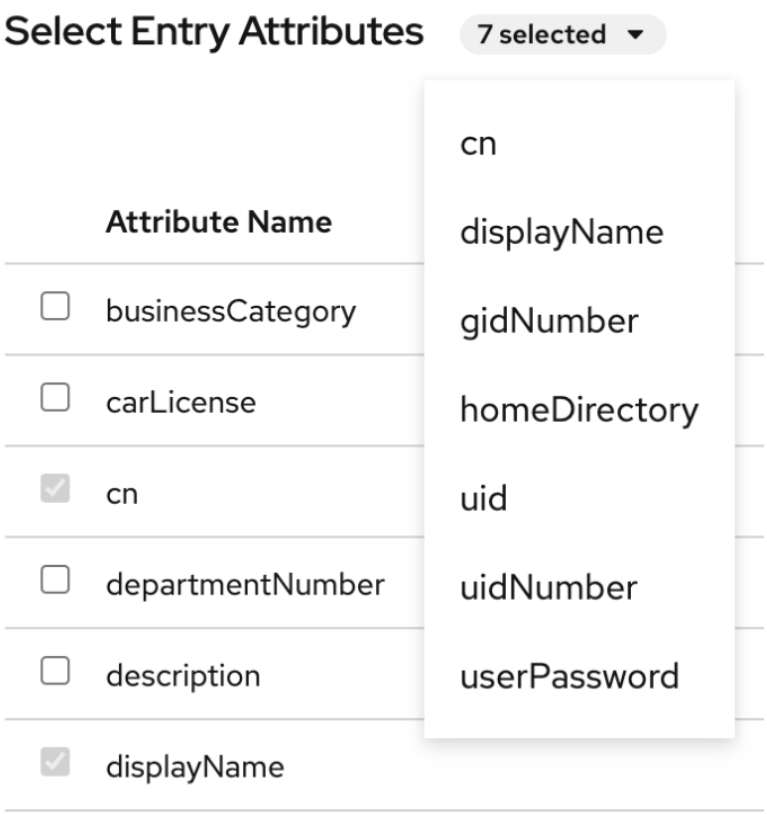
Procedure

1. Open the **LDAP Browser** menu to reveal the list of existing suffixes.
2. Using the **Tree** or **Table** view, expand the parent entry **ou=people,dc=example,dc=com** under which you want to create a user.
3. Click the Options menu (▮) and select **New** to open the wizard window.



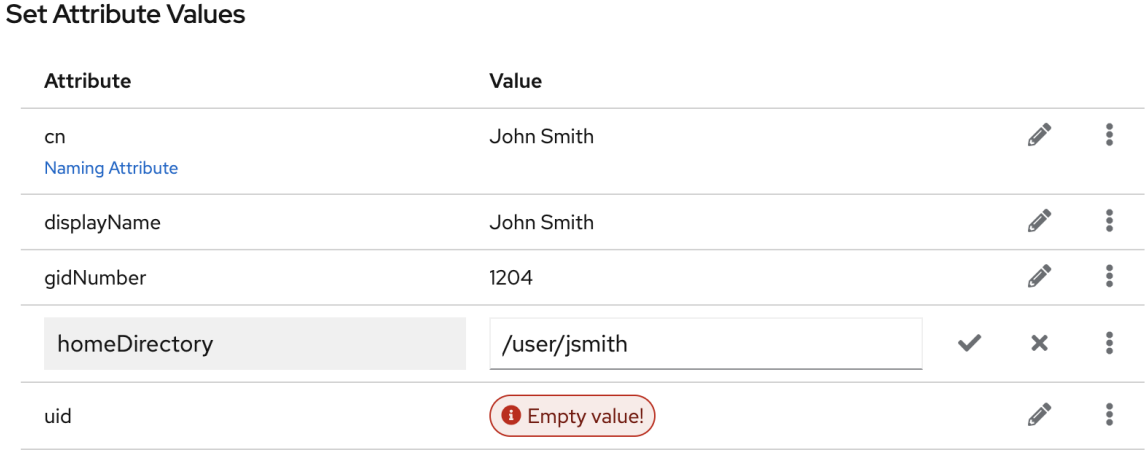
4. Select the **Create a new User** option and click **Next**.
5. For the user entry, select **Posix Account** type and click **Next**.

6. Optional: Select additional attributes, such as **userPassword**, and click **Next**. You can view all selected attributes by expanding the drop-down list near the step name.



7. Set a value for each attribute:

   a. Click on the pencil button of the attribute and add a value.



   Note that a separate menu opens when you set the **userPassword** value. The value is filled with asterisks (*) to hide the plain text.

   b. Click on the check button to save changes.

   c. Optional: Set an additional attribute value by clicking the **Options menu (⋮) → Add Another Value**.

   d. After you have set all values, click **Next**.

8. Verify that all entry details are correct and click **Create User**. Directory Server creates the entry with mandatory attributes for a POSIX user and sets the password to it. You can click **Back** to modify entry settings, or click **Cancel** to cancel the entry creation.

9. View the **Result for Entry Creation** and click **Finish**.

Verification

1. Navigate to **LDAP Browser → Search**.
2. Select the database suffix that contains the entry, such as **dc=example,cd=com**.
3. Enter your search criteria in the field, such as **John**, and press **Enter**.
4. Find the entry you recently created in the list of entries.

2.2. EDITING AN LDAP ENTRY USING THE WEB CONSOLE

You can modify a directory entry using the web console. This example modifies a user entry **cn=John Smith,ou=people,dc=example,dc=com** by:

- adding telephone numbers **556778987** and **556897445**.
- adding email **jsmith@example.com**.
- changing the password.

Prerequisites

- You are logged into the Directory Server web console.

Procedure

1. Open the **LDAP Browser** menu.
2. Using the **Tree** or **Table** view, expand the entry you want to edit, such as **cn=John Smith,ou=people,dc=example,dc=com**.
3. Click the Options menu (▮) and select **Edit** to open the wizard window.
4. Optional: In the **Select ObjectClasses** step, add or delete object classes for the entry. Click **Next**.
5. In the **Select Attributes** step, add **telephoneNumber** and **mail** attributes to the entry and click **Next**. If you do not see an attribute you want to add to the entry it means that you did not add corresponding object class in the previous step.

NOTE

In this step, you **can not** delete mandatory attributes of the selected object classes.

6. In the **Edit Attribute Values** step, set **telephoneNumber** to **556778987** and **556897445**, **mail** to **jsmith@example.com** and change **userPassword** value:
   a. Click on the pencil button of the attribute and add or change a new value.
   b. Click on the check button to save changes.

c. Optional: Set an additional value to an attribute by clicking the **Options menu (▮) → Add Another Value**. The **telephoneNumber** attribute has two values in this example. When you set all values, click **Next**.

7. Review your changes and click **Next**.
8. To edit the entry, click **Modify Entry**. You can click **Back** to make other changes to the entry, or click **Cancel** to cancel the entry editing.
9. View the **Result for Entry Modification** and click **Finish**.

Verification

- Expand the entry details and view the new changes appear among the entry attributes.

Additional resources

- Using roles in Directory Server
- Using groups in Directory Server

2.3. RENAMING AND RELOCATING AN LDAP ENTRY OR SUBTREE USING THE WEB CONSOLE

You can rename or relocate a directory entry or a subtree using the web console. This example renames and relocates the entry **cn=John Smith,ou=people,dc=example,dc=com** to **cn=Tom Smith,ou=clients,dc=example,dc=com**.

Prerequisites

- You are logged into the Directory Server web console.

Procedure

1. Open the **LDAP Browser** menu.
2. Using the **Tree** or **Table** view, expand the entry you want to modify, such as **cn=John Smith,ou=people,dc=example,dc=com**.
3. Click the Options menu (▮) and select **Rename** to open the wizard window.
4. In the **Select The Naming Attribute And Value** step:
   a. Set a new value **Tom Smith** for the naming attribute **cn** and click **Next**.
   b. Optional: Select another naming attribute from the drop-down menu.
   c. Optional: In case you want to delete the old entry and create a new one using the new RDN, check the **Delete the old RDN**
5. In **Select The Entry Location** step, select the parent entry for the new location, and click **Next**.
6. Review changes you made to the entry and click **Next**.
7. If the entry details are correct, click **Change Entry Name**. You can click **Back** to make other changes to the entry or click **Cancel** to cancel the entry modification.

8. View **Result for Entry Modification** and click **Finish**.

Verification

- Expand the entry details and review the updated entry.

2.4. DELETING AN LDAP ENTRY USING THE WEB CONSOLE

You can delete a directory entry or a subtree using the web console. This example deletes the entry **cn=Tom Smith,ou=clients,dc=example,dc=com**.

Prerequisites

- You are logged into the Directory Server web console.

Procedure

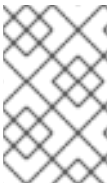
1. Open the **LDAP Browser** menu.
2. Using the **Tree** or **Table** view, expand the entry you want to delete, such as **cn=Tom Smith,ou=people,dc=example,dc=com**.
3. Click the Options menu (▮) and select **Delete** to open the wizard window.
4. Click **Next** after you review the data about the entry you want to delete.
5. In the **Deletion** step, toggle the switch to the **Yes, I'm sure** position and click **Delete**. You can click **Cancel** to cancel the entry deletion.
6. View the **Result for Entry Deletion** and click **Finish**.

Verification

1. Navigate to **LDAP Browser → Search**.
2. Select the suffix where the entry previously existed, such as **dc=example,cd=com**.
3. Enter your search criteria in the field, such as **Tom**, and press **Enter**.
4. Verify that the deleted entry is no longer present.

CHAPTER 3. ASSIGNING AND MANAGING UNIQUE NUMERIC ATTRIBUTE VALUES

Some entry attributes require unique numeric identifiers, such as **uidNumber** and **gidNumber**. The Directory Server can generate and assign these unique numbers automatically for specified attributes using the Distributed Numeric Assignment (DNA) Plug-in.



NOTE

The DNA plug-in does not guarantee *attribute uniqueness*. The pug-in allocates non-overlapping ranges, enabling manual assignment of numbers to managed attributes without mandating or verifying their uniqueness.

With DNA plug-in, you can effectively avoid replication conflicts. The DNA Plug-in assigns unique numbers across a *single* backend. For multi-supplier replication, when each supplier is running a local DNA plug-in instance, you must assign different ranges of numbers to each server. This ensures that each instance is using a truly unique set of numbers.

3.1. ABOUT DYNAMIC NUMBER ASSIGNMENTS

The DNA plug-in assigns a range of available numbers that instance can issue. Two attributes define the range definition: the server next available number (the botton value of the range) and its maximum value (the upper value of the range). You set the initial bottom value when you configure the plug-in. Later, the plug-in udates this bottom value.

By breaking the available numbers into separate ranges on each replica, the servers can continually assign numbers without overlapping with each other.

3.1.1. Filters, searches, and target entries

The server performs a sorted search internally to verify if another server has already taken the next specified range, requiring the managed attribute to have an equality index with the proper ordering matching rule.

The DNA Plug-in is always applied to a specific area of the directory tree (the *scope*) and specific entry types within that subtree (the *filter*).

IMPORTANT

The DNA Plug-in works only on a *single* back end, unable to manage number assignments for multiple databases. The DNA Plug-in uses the sort control to check whether a value has been manually allocated outside of the DNA Plug-in. However, this validation using the sort control works only on a *single* back end.

3.1.2. Ranges and assigning numbers

The Directory Server can generate attribute values using several different methods:

- In a basic scenario, when adding a user entry to the directory with an object class that requires the unique-number attribute but doesn't have the attribute value, it activates the DNA Plug-in to assign a value. This occurs when the DNA Plug-in is configured to assign unique values to a single attribute.

- A simpler option entails using a *magic number* as a template value for the managed attribute. This magic number, which can be a number or even a word, resides outside the server's range. The plug-in recognizes it as a signal to replace it with a newly assigned value. When an entry is added with the magic value and falls within the scope and filter of the configured DNA Plug-in, it prompts the plug-in to generate a new value. For instance, using **ldapmodify**, you can add 0 as a magic number:

```
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: posixAccount
uid: jsmith
cn: John Smith
uidNumber: 0
gidNumber: 0
```

The DNA Plug-in only generates new, unique values. If an entry is added or modified to use a specific value for an attribute controlled by the DNA Plug-in, the plug-in will not overwrite it.

3.1.3. Multiple attributes in the same range

The DNA Plug-in can assign unique numbers to a single or multiple attribute types from a single range of unique numbers.

This offers multiple options for assigning unique numbers to attributes:

- A single number for a single attribute type from a unique range.
- The same unique number for two attributes in one entry.
- Two distinct attributes assigned distinct numbers from the same unique range.

In many cases, assigning a unique number per attribute type suffices. For instance, when assigning an **employeeID** to a new employee entry, it's crucial to ensure each employee entry receives a unique **employeeID**.

However, in some cases, it's useful to assign unique numbers from the same range of numbers to multiple attributes. For instance, when assigning a **uidNumber** and a **gidNumber** to a **posixAccount** entry, the DNA Plug-in assigns the same number to both attributes. To achieve this, pass both managed attributes to the modify operation specifying the magic value. Using **ldapmodify**:

```
#  ldapmodify -D "cn=Directory Manager" -W -x

dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
-
add:gidNumber
gidNumber: 0
```

When the DNA Plug-in handles multiple attributes, it can assign a unique value to only one attribute if the object class permits only one. For instance, the **posixGroup** object class allows **gidNumber** but not **uidNumber**. If the DNA Plug-in manages both **uidNumber** and **gidNumber**, it assigns a unique number

for **gidNumber** from the **uidNumber** and **gidNumber** attribute range when creating a **posixGroup** entry. Sharing a pool for all managed attributes ensures consistent assignment of unique numbers, preventing conflicts where **uidNumber** and **gidNumber** on different entries end up with the same number from separate ranges.

If the DNA Plug-in manages multiple attributes, it assigns the same value to all of them in a single modify operation. To assign *different* numbers from the same range, you need to perform separate modify operations. For example, you can use **ldapmodify**:

```
# ldapmodify -D "cn=Directory Manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
^D

# ldapmodify -D "cn=Directory Manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: employeeId
employeeId: magic
```

IMPORTANT

To assign unique numbers to multiple attributes using the DNA Plug-in, specify a unique value for each attribute that needs it. Unlike a single attribute, which doesn't require this, multiple attributes need you to specify the unique value. In some cases, an entry won't permit all attributes in the range, or it may allow all types but only a subset requiring a unique value.

Example 3.1. Example. DNA and Unique Bank Account Numbers

Example Bank administrator configures the DNA Plug-in to assign a shared unique number to a customer's **primaryAccount** and **customerID** attributes.

The bank also wants to assign unique numbers for secondary accounts, distinct from primary accounts but from the same number range as the customer IDs and primary accounts. The Example Bank administrator configures the DNA Plug-in to manage the **secondaryAccount** attribute, added post-entry creation, after assigning unique numbers to **primaryAccount** and **customerID**. This guarantees a shared unique number for **primaryAccount** and **customerID**, with distinct and unique **secondaryAccount** numbers from the same range.

CHAPTER 4. ENFORCING ATTRIBUTE UNIQUENESS

To ensure that the value of an attribute is unique across the whole directory or a subtree, you can use the Attribute Uniqueness plug-in, which is disabled by default.

You can configure the plug-in to verify attribute uniqueness either of the following ways:

- Set a list of subtrees where the plug-in must check attribute uniqueness by using the **uniqueness-subtrees** parameter, for example:

  ```
  uniqueness-attribute-name: mail
  uniqueness-subtrees: ou=accounting,dc=example,dc=com
  uniqueness-subtrees: ou=sales,dc=example,dc=com
  uniqueness-across-all-subtrees: on
  uniqueness-exclude-subtrees: ou=private,ou=people,dc=example,dc=com
  ```

  For more details, see Configuring the Attribute Uniqueness plug-in over subtrees .

- Set a parent entry object class by using the **uniqueness-top-entry-oc** parameter. If a parent entry of the updated entry contains this object class then the plug-in checks for the uniqueness of attributes under the parent entry subtree. For example, you can configure the plug-in the following way:

  ```
  uniqueness-attribute-name: mail
  uniqueness-top-entry-oc: nsContainer
  uniqueness-subtree-entries-oc: inetOrgPerson
  uniqueness-exclude-subtrees: ou=private,ou=people,dc=example,dc=com
  ```

  For more details, see Configuring the Attribute Uniqueness plug-in over object classes .

You can create multiple configuration entries of the plug-in to apply different conditions. Directory Server stores all configuration entries of the plug-in under **cn=plugins,cn=config**.

4.1. CONFIGURING THE ATTRIBUTE UNIQUENESS PLUG-IN OVER SUBTREES USING THE COMMAND LINE

You can use the **dsconf** utility to set the list of subtrees where the plug-in must check the attribute uniqueness. A subtree can be any entry in the directory, including a suffix.

Use the following example procedure to configure the plug-in to verify uniqueness of the **mail** attribute in entries under the **ou=sales,dc=example,dc=com** and **ou=accounting,dc=example,dc=com** subtrees.

Prerequisites

- You have the Directory Manager permissions.

Procedure

1. Create a new plug-in configuration entry:

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq add
   "Mail Uniqueness" --attr-name mail --subtree ou=sales,dc=example,dc=com
   ou=accounting,dc=example,dc=com
   ```

The command creates the **cn=Mail Uniqueness,cn=plugins,cn=config** configuration entry.

NOTE

You can set the plug-in to verify uniqueness of multiple attributes in one configuration entry.

2. Optional: Configure uniqueness across all subtrees configured in this plug-in configuration entry:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set
"Mail Uniqueness" --across-all-subtrees on
```

The command sets the **uniqueness-across-all-subtrees** plug-in configuration parameter to **on**. Therefore, the plug-in checks uniqueness of the **mail** attribute across both the **ou=sales,dc=example,dc=com** and **ou=accounting,dc=example,dc=com** subtrees. By default, the plug-in only checks uniqueness across the subtree where the entry is created or updated, which means that if you create or update an entry under **ou=sales,dc=example,dc=com**, the plug-in checks the **mail** attribute uniqueness only across this subtree.

3. Optional: Set a subtree that the plug-in must exclude from the attribute uniqueness verification.
For example, if you want the plug-in to skip the **ou=internal,ou=sales,dc=example,dc=com** subtree, you can use the **ldapmodify** utility to set the **uniqueness-exclude-subtrees** parameter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=Mail Uniqueness,cn=plugins,cn=config
changetype: modify
add: uniqueness-exclude-subtrees
uniqueness-exclude-subtrees: ou=internal,ou=sales,dc=example,dc=com
```

4. Optional: If you what the plug-in to verify uniqueness only in entries that contain a specific object class, set this object class as a value for the **uniqueness-subtree-entries-oc** parameter.
For example, you want the **mail** attribute to be unique only in entries that contain the **inetOrgPerson** object class, enter:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set
"Mail Uniqueness" --subtree-entries-oc=inetOrgPerson
```

5. Enable the plug-in on the server:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq enable
"Mail Uniqueness"
```

6. Restart the instance:

```
# dsctl instance_name restart
```

Verification

- View the configuration entry details:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq show
"Mail Uniqueness"

dn: cn=Mail Uniqueness,cn=plugins,cn=config
cn: Mail Uniqueness
nsslapd-plugin-depends-on-type: database
nsslapd-pluginDescription: Enforce unique attribute values
nsslapd-pluginEnabled: on
...
uniqueness-across-all-subtrees: on
uniqueness-attribute-name: mail
uniqueness-exclude-subtrees: ou=internal,ou=sales,dc=example,dc=com
uniqueness-subtree-entries-oc: inetOrgPerson
uniqueness-subtrees: ou=accounting,dc=example,dc=com
uniqueness-subtrees: ou=sales,dc=example,dc=com
```

Additional resources

- Attribute Uniqueness plug-in attributes

4.2. CONFIGURING THE ATTRIBUTE UNIQUENESS PLUG-IN OVER OBJECT CLASSES

You can configure the Attribute Uniqueness plug-in to ensure that values of an attribute are unique in entries that contain a specific object class. To configure the plug-in, you must set the following configuration parameters:

- **uniqueness-top-entry-oc**. This parameter uniquely identifies a subtree under which the plug-in verifies attribute uniqueness. The plug-in verifies uniqueness only in entries those parent entries contain the specific object class you set in **uniqueness-top-entry-oc**. If Directory Server did not find the object class in the parent entry of the updated entry, the search continues at the next higher level entry up to the root of the directory tree.

- **uniqueness-subtree-entries-oc**. This parameter identifies which entries the plug-in must check. When you set an object class in the **uniqueness-subtree-entries-oc** parameter, the plug-in verifies uniqueness of attributes only in updated entries that contain this specific object class.

Use the following example procedure to set the **mail** attribute to be unique in all entries under the entry that contains the **nsContainer** object class set and for the plug-in to search the **mail** attribute in entries that contain the **inetOrgPerson** object class.

Prerequisites

- You have the Directory Manager permissions.

Procedure

1. Create a new plug-in configuration entry:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq add
"Mail Uniqueness with OC" --attr-name mail --subtree-entries-oc=inetOrgPerson --top-
entry-oc=nsContainer
```

The command creates the **cn=Mail Uniqueness with OC,cn=plugins,cn=config** entry with the configured **uniqueness-top-entry-oc** and **uniqueness-subtree-entries-oc** plug-in parameters.

2. Optional: Set a subtree that the plug-in must exclude from the attribute uniqueness verification.
   For example, if you want the plug-in to skip the **ou=internal,ou=sales,dc=example,dc=com** subtree, use the **ldapmodify** utility to set the **uniqueness-exclude-subtrees** parameter:

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=Mail Uniqueness with OC,cn=plugins,cn=config
changetype: modify
add: uniqueness-exclude-subtrees
uniqueness-exclude-subtrees: ou=internal,ou=sales,dc=example,dc=com
```

3. Enable the plug-in on the server:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq enable
"Mail Uniqueness with OC"
```

4. Restart the instance:

```
# dsctl instance_name restart
```

Verification

- View the configuration entry details:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq show
"Mail Uniqueness with OC"

dn: cn=Mail Uniqueness with OC,cn=plugins,cn=config
cn: Mail Uniqueness with OC
nsslapd-plugin-depends-on-type: database
nsslapd-pluginDescription: none
nsslapd-pluginEnabled: on
...
uniqueness-attribute-name: mail
uniqueness-exclude-subtrees: ou=internal,ou=sales,dc=example,dc=com
uniqueness-subtree-entries-oc: inetOrgPerson
uniqueness-top-entry-oc: nsContainer
```

4.3. CONFIGURING THE ATTRIBUTE UNIQUENESS PLUG-IN USING THE WEB CONSOLE

You can use the web console to configure the Attribute Uniqueness plug-in. Note that you can create different configuration entries of the plug-in to apply different conditions.

Use the following example procedure to configure the plug-in to verify uniqueness of the **mail** attribute in entries under the **ou=sales,dc=example,dc=com** and **ou=accounting,dc=example,dc=com** subtrees.

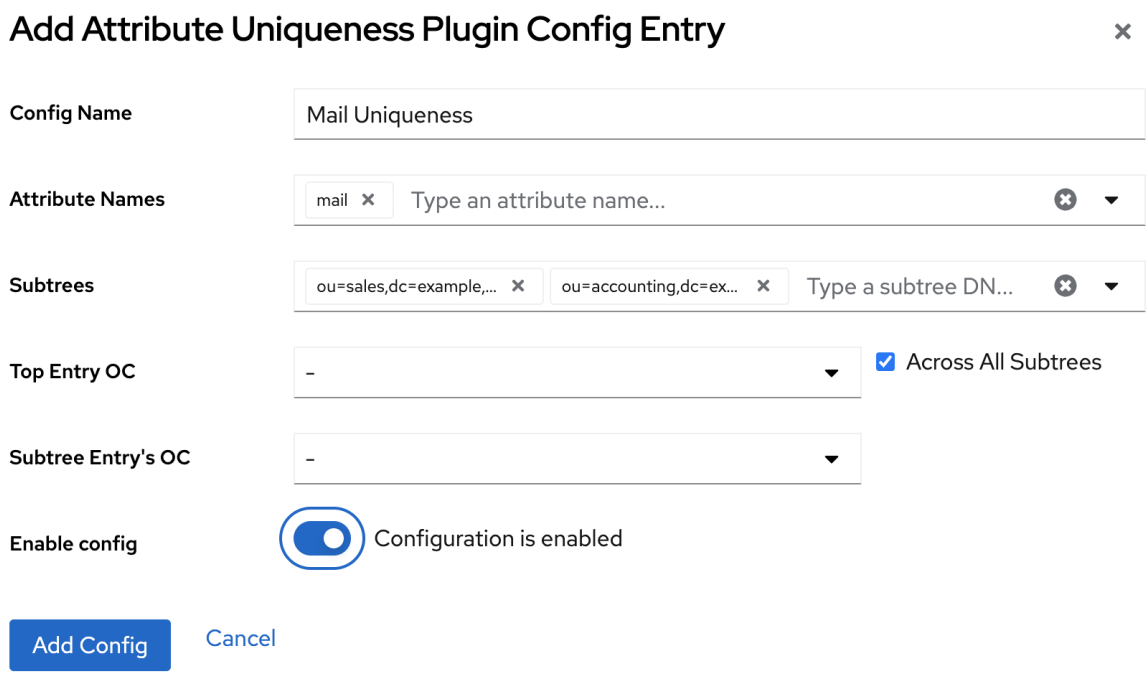
Prerequisites

- You have the Directory Manager permissions.
- You are logged in to the web console. For more details, see Logging in to the Directory Server by using the web console.

Procedure

1. Select the instance, where you want to configure the plug-in.
2. Open the **Plugins** menu and select the **Attribute Uniqueness** plug-in from the list.
3. Click **Add Config** button to start the configuration of a new configuration entry.
4. Enter the name of the configuration entry in the **Config Name** field.
5. Select which attributes must be unique in the **Attribute Names** field. The field sets the **uniqueness-attribute-name** attribute.
6. Enter the subtrees under which the plug-in checks uniqueness of attributes in the **Subtrees** field. The field sets the **uniqueness-subtrees** attribute.
   By default, the plug-in checks uniqueness across only the subtree where the entry is created or updated. To check across all listed subtrees, check the **Across All Subtrees** checkbox that sets the **uniqueness-across-all-subtrees** attribute to **on**.
7. Toggle the switch to the **Configuration is enabled** position.
8. Click **Add Config** button to create the plug-in configuration entry.

Figure 4.1. Configuration example of the Attribute Uniqueness plug-in.



9. Restart the instance. For more details, see Starting and stopping a Directory Server instance by using the web console.

Verification

- Find the newly created plug-in entry in the list of configuration entries.

Additional resources

- Attribute Uniqueness plug-in attributes