LDAP Setup on podman

Requirement

1. Install and configure Ldap 389 DS in the containerised platform with persistence storage
2. Create organisation keenable.in
3. Create OU [Dev,Support,POC, Document, Observability]
4. Create group Admin,Support
5. Create custom attribute

Environment Info

Server Info:-

Os version

NAME="Ubuntu"

VERSION="20.04.6 LTS (Focal Fossa)"

podman version 3.4.2

Client Info:-

NAME="Ubuntu"

VERSION="20.04.6 LTS (Focal Fossa)"

Ldap version:- 3

Apache Directory Studio version:- 2.0.0.v20210717-M17

List of Tools:-

1. Podman
2. Ldap
3. Apache Directory Studio

LDAP:- The Lightweight Directory Access Protocol is a communication protocol used to access directory servers and is used to store, update and retrieve data from a directory structure.

command for the setup or configuration

create directory with name 389ds and also create directory data inside

```
mkdir -p 389ds/data
```

```
cd 389ds/data
```

1. Create Bash script for create Pod and container

vim ldap.sh

Bash script :-

```
#!/bin/bash
#create pod with name ldap389
podman pod create --name ldap389 --publish 3389:3389 --publish 3636:3636
#create container
podman run -dt \
--pod ldap389 \
--name 389ds-ldap \
-v /home/ravi/389ds/data:/data \
-e DS_SUFFIX=dc=keenable,dc=in \
-e DS_DM_PASSWORD=ravi \
docker.io/389ds/dirsrv
```

- **ldap389** Created a pod named
- **#!/bin/bash:** This is called a shebang and specifies that the script should be executed using the Bash shell.
- **podman pod create --name ldap389 --publish 3389:3389 --publish 3636:3636:**
- **podman:** This is the command-line tool used to manage pods and containers.
- **pod create:** This subcommand is used to create a new pod.
- **-name ldap389:** This option specifies the name of the pod as "ldap389".
- **-publish 3389:3389 --publish 3636:3636:** These options publish the specified container ports to the host. Ports 3389 and 3636 are mapped from the host to the pod. This allows services running inside the pod to be accessed from the host using these ports.
- **podman run -dt:** This command runs a container.
  - **d:** Detaches the container from the terminal, allowing it to run in the background.

    - **t:** Allocates a pseudo-TTY, which helps with handling input/output.
- **-pod ldap389:** This option specifies that the container should be created within the "ldap389" pod that was created earlier.
- **-name 389ds-ldap:** This sets the name of the container to "389ds-ldap".
- **v ~/389ds/data:/data:** This mounts the directory **~/389ds/data** from the host into the container at the **/data** directory. This is used for data persistence, allowing data to be stored on the host filesystem.
- **e DS_SUFFIX=dc=keenable,dc=in:** This sets the environment variable **DS_SUFFIX** within the container to specify the LDAP directory suffix.
- **e DS_DM_PASSWORD=:** This sets the environment variable **DS_DM_PASSWORD** within the container to specify the password for the Directory Manager of the LDAP server.
- **docker.io/389ds/dirsrv:** This is the Docker image that will be used to create the container. It's the 389 Directory Server image from Docker Hub.

In Pod we have set port 3389 and 3636

In container we have set base Dn as keenable.in and set password of dn

- Give permmision to script file

```
chmod 777 ldaptest.sh
```

> changes the permissions of the file and permissions to 777 means that the owner, the group, and everyone else can read, write, and execute the script.

- Run the script

```
sh -x ldaptest.sh
```

- Check container list

```
podman ps -a --pod
```

- **Install Ldap utility on bash machine for runn ldap command**

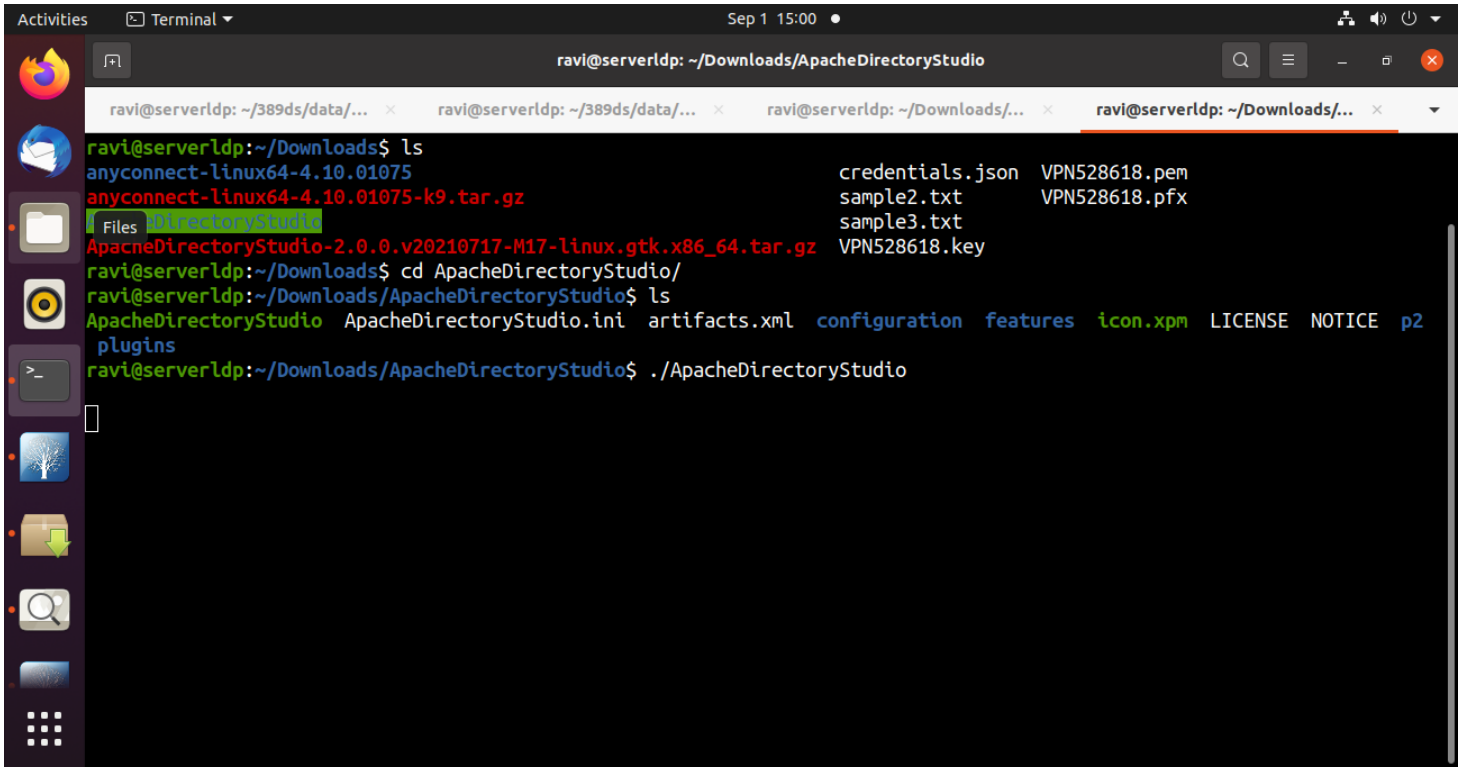
```
sudo apt install ldap-utils
```

- **sudo:** This command is used to run with superuser (root) privileges.
- **apt:** It's used to manage software packages, including installation, updating, and removal.

- **install:** This is an argument passed to the apt command, indicating that you want to install a package.
- **ldap-utils:** This is a collection of command-line utilities for interacting with LDAP (Lightweight Directory Access Protocol) servers. These utilities are useful for managing and querying directory services.

2. Setup ApacheDirectory studio for ldap db UI

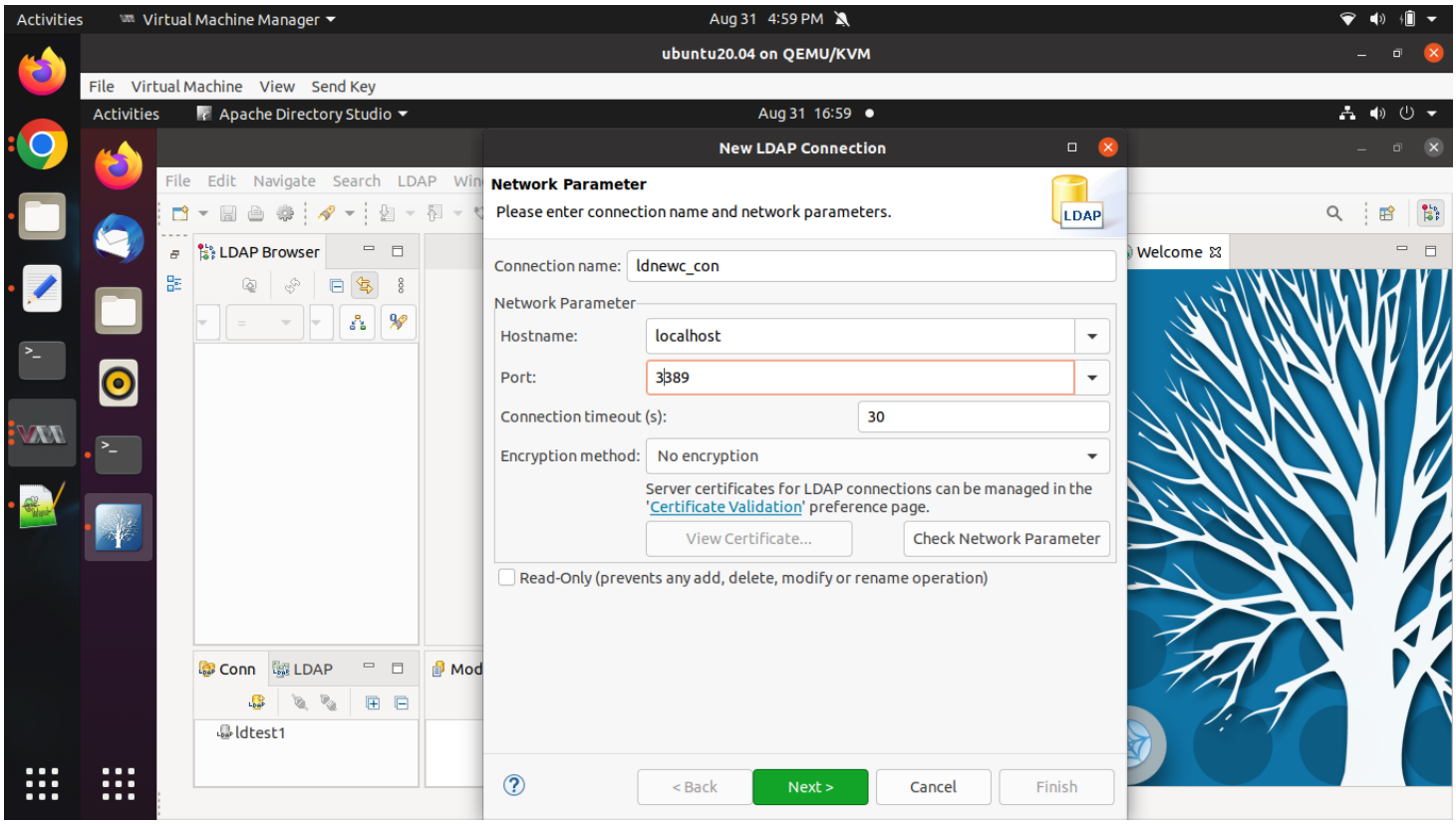
a. Install ApacheDirectory studio tar file and extract in directory

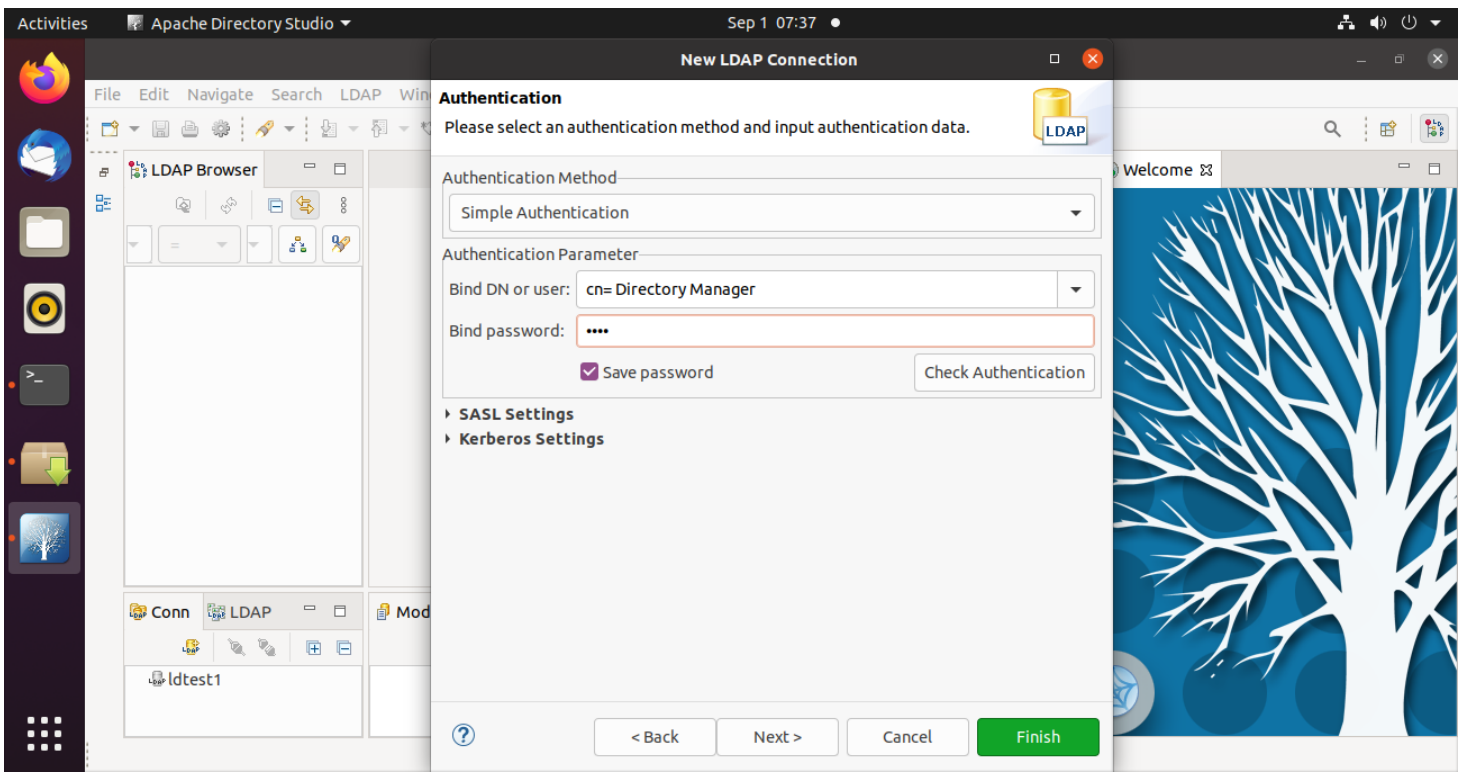


b. open apache directory

c. Go to new connection and enter some details

```
ldap connection name
Hostname -> localhost
ldap port > Press next (3389)
Enter Bind DN name (cn= Directory Manager)
Enter Bind DN password -> finish
```





Create Organization_unit ldif file (file extension name , ldif)

```
cat  organisation.ldif
```

1. we have create 5 organisation (Dev, Support, POC, Document, Observability)

```
dn: dc=keenable,dc=in
objectClass: top
objectClass: domain
dc: keenable

dn: ou=Dev,dc=keenable,dc=in
objectClass: top
objectClass: organizationalUnit
ou: Dev

dn: ou=Support,dc=keenable,dc=in
objectClass: top
objectClass: organizationalUnit
ou: Support

dn: ou=POC,dc=keenable,dc=in
objectClass: top
objectClass: organizationalUnit
ou: POC

dn: ou=Document,dc=keenable,dc=in
objectClass: top
objectClass: organizationalUnit
ou: Document

dn: ou=Observability,dc=keenable,dc=in
objectClass: top
objectClass: organizationalUnit
ou: Observability
```

Run this command to add **organisation.ldif**

```
ldapadd -a -c -x -H ldap://localhost:3389 -D "cn=Directory Manager" -W  -f organisation.
```

- **ldapadd:** This command is used to add new entries to an LDAP directory server.
- **a:** It's used for adding new data without overwriting existing data.
- **c**:If the LDAP server has a schema, this option helps ensure that the data conforms to the defined schema.
- **x:** This option specifies that simple authentication should be used. Similar to -**x** in **ldapsearch**, it's used for quick testing,
- **H ldap://localhost:3389:** This option specifies the LDAP URI (Uniform Resource Identifier) of the LDAP server. It's connecting to the LDAP server running on the local machine (**localhost**) at port **3389.**
- **D "cn=Directory Manager":** This option specifies the Bind DN (Distinguished Name) to authenticate with the LDAP server. In this case, it's using the "Directory Manager" account to

authenticate.

- **W:** This option prompts you to enter the password for the Bind DN interactively after providing the -D option. It's safer than putting the password in the command line itself.
- **f organisation.ldif**: This option specifies the name of the LDIF (LDAP Data Interchange Format) file that contains the data to be added to the LDAP server. The **f** option is followed by the filename **(organisation.ldif** in this case).

4. Create 2 group inside Support

```
vim  group.ldif


dn: cn=Admins,ou=Support,dc=keenable,dc=in
objectClass: top
objectClass: groupOfUniqueNames
cn: Admins
uniqueMember: uid=user1,ou=Support,dc=keenable,dc=in

dn: cn=SupportTeam,ou=Support,dc=keenable,dc=in
objectClass: top
objectClass: groupOfUniqueNames
cn: SupportTeam
uniqueMember: uid=user2,ou=Support,dc=keenable,dc=in
```

Run this command to add **group.ldif**

```
ldapadd -a -c -x -H ldap://localhost:3389 -D "cn=Directory Manager" -W -f group.ldif
```

- **W:** This option prompts you to enter the password for the Bind DN interactively after providing the **D** option. It's safer than putting the password in the command line itself.
- **f group.ldif:** This option specifies the name of the LDIF (LDAP Data Interchange Format) file that contains the data to be added to the LDAP server. The **f** option is followed by the filename (**group.ldif** in this case).

5. Run some Command of ldap

a. First check how many default object class created

```
ldapsearch -o ldif-wrap=no -x -H ldap://localhost:3389 -D "cn=Directory Manager" -w "rav
```

- **ldapsearch:** This command is used to search and retrieve information from an LDAP directory server.

- **o ldif-wrap=no:** This option turns off LDIF (LDAP Data Interchange Format) line wrapping. It ensures that the output LDIF is not formatted with line breaks, which can be useful for processing the output programmatically.
- **D "cn=Directory Manager"**: This option specifies the Bind DN (Distinguished Name) to authenticate with the LDAP server. In this case, it's using the "Directory Manager" account to authenticate.
- **w "redhat@"**: This option specifies the password for the Bind DN. The password is provided directly in the command line using double quotes.
- **b "cn=schema"**: This option specifies the base DN (Distinguished Name) from which the search should start. It's set to **"cn=schema"** which indicates that the search should start from the schema entry.
- **'(objectClass=subSchema)**': This is the search filter. It specifies that the search should retrieve entries with the **objectClass** attribute equal to **subSchema**. This filter targets the schema definition entries.
- **s sub**: This option sets the search scope. In this case, it's set to "sub" which means a subtree search, i.e., it searches for the specified filter under the specified base DN and all its subordinates.
- **objectclasses:** This is the attribute(s) we want to retrieve from the entries that match the search filter. In this case, we're requesting the **objectclasses** attribute.

a. Check how many default attributes created

```
ldapsearch -o ldif-wrap=no -x -H ldap://localhost:3389 -D "cn=Directory Manager" -w "rec
```

- **o ldif-wrap=no:** This option turns off LDIF (LDAP Data Interchange Format) line wrapping. It ensures that the output LDIF is not formatted with line breaks, which can be useful for processing the output programmatically.
- **D "cn=Directory Manager":** This option specifies the Bind DN (Distinguished Name) to authenticate with the LDAP server. In this case, it's using the "Directory Manager" account to authenticate.
- **b "cn=schema"**: This option specifies the base DN (Distinguished Name) from which the search should start. It's set to **"cn=schema"** which indicates that the search should start from the schema entry.
- **'(objectClass=subSchema)'**: This is the search filter. It specifies that the search should retrieve entries with the **objectClass** attribute equal to **subSchema.** This filter targets the schema definition entries.
- **attributetypes:** This is the attribute(s)we want to retrieve from the entries that match the search filter. In this case, we 're requesting the **attributetypes** attribute.

6. Create Custom attribute according to our requirement

a. Create customer attribute ldif file

```
vim  custom_attribute.ldif
```

```
dn: cn=schema
changetype: modify
add: attributeTypes
attributetypes: (emp_code-oid NAME  'EmployeeCode' DESC 'EmployeeCode' EQUALITY caseIgnc
attributetypes: (gender-oid NAME 'Gender' DESC 'Gender' EQUALITY caseIgnoreMatch SUBSTR
attributetypes: (certifications-oid NAME 'Certifications' DESC 'Certifications' EQUALITY
attributetypes: (passport-oid NAME 'PassportNo'  DESC 'PassportNo.' EQUALITY caseIgnoreM
attributetypes: (pan_no-oid NAME 'Panno' DESC 'Panno.' EQUALITY caseIgnoreMatch SUBSTR c
attributetypes: (qualification-oid NAME 'Qualification' DESC 'Qualification' EQUALITY ca
attributetypes: (correspondence_address-oid  NAME 'CorrespondenceAddress' DESC 'Correspc
attributetypes: (personalemail-id-oid NAME 'personalemail-id' DESC 'personalemail-id' EQ
attributetypes: (facebookaccount-oid NAME 'facebookaccount' DESC 'facebookaccount' EQUAL
attributetypes: (twitteraccount-oid NAME 'twitteraccount' DESC 'twitteraccount' EQUALITY
attributetypes: (MaritalStatus-oid NAME 'MaritalStatus' DESC 'MaritalStatus' EQUALITY ca
attributetypes: (Childinfo-oid NAME 'Childinfo' DESC 'Childinfo' EQUALITY caseIgnoreMatc
attributetypes: (pfno-oid NAME 'pfno' DESC 'pfno' EQUALITY caseIgnoreMatch SUBSTR caseEx
attributetypes: (bankname-oid NAME 'BankName' DESC 'BankName' EQUALITY caseIgnoreMatch S
attributetypes: (AccountNo-oid NAME 'AccountNo' DESC 'AccountNo' EQUALITY caseIgnoreMatc
attributetypes: (IFSCCode-oid  NAME 'IFSCCode' DESC 'IFSCCode' EQUALITY caseIgnoreMatch
attributetypes: (ESICCardNo-oid NAME 'ESICCardNo' DESC 'ESICCardNo' EQUALITY caseIgnoreM
attributetypes: (FamilyMembersInsured-oid  NAME 'FamilyMembersInsured' DESC 'FamilyMembe
attributetypes: (documentssubmitted-oid  NAME 'documentssubmitted' DESC 'documentssubmit
attributetypes: (doj-oid NAME 'DateofJoining' DESC 'DateofJoining' EQUALITY caseIgnoreIA
attributetypes: (doreg-oid NAME 'DateOfResignation' DESC 'DateOfResignation' EQUALITY ca
attributetypes: (dob-oid NAME 'DateofBirth' DESC 'DateofBirth' EQUALITY caseIgnoreIA5Mat
attributetypes: (passport_valid_upto-oid NAME 'PassportValidupto' DESC 'PassportValidupt
attributetypes: (ProfessionalStartYEARS-oid NAME 'ProfessionalStartYEARS' DESC 'Professi
attributetypes: (YEARSOfExperience-oid NAME 'YEARSOfExperience' DESC 'YEARSOfExperience'
attributetypes: (dateofjoiningasintern-oid NAME 'dateofjoiningasintern' DESC 'dateofjoir
attributetypes: (AadharNo-oid  NAME 'AadhaarNo' DESC 'AadhaarNo' EQUALITY integerMatch S
attributetypes: (YearsofQualification NAME 'YearsofQualification' DESC 'YearsofQualifica
attributetypes: (mobileno-oid  NAME 'mobileno' DESC 'mobileno' EQUALITY integerMatch SYN
attributetypes: (UANno-oid NAME 'UANno' DESC 'UANno' EQUALITY integerMatch SYNTAX 1.3.6.
attributetypes: (InsuranceMonthlyAmountDeductionINR-oid NAME 'InsuranceMonthlyAmountDedu
attributetypes: ( projectname-oid NAME 'ProjectName' DESC 'ProjectName' EQUALITY caseIgr
```

b. Add this file to ldap db

```
ldadadd -a -c -x -H ldap://localhost:3389 -D "cn=Directory Manager" -W  -f custom_attrib
```

c. Create **Object class** file for add attribute to object class

```
vim  object_class.ldif
```

```
dn: cn=schema
changetype: modify
add: objectClasses
objectClasses: ( customEmployee-oid NAME 'customEmployee' SUP top STRUCTURAL MUST ( Empl
```

d. Add object class ldif file

```
ldapadd -a -c -x -H ldap://localhost:3389 -D "cn=Directory Manager" -W -f object_class.l
```

- **f object_class.ldif:** This option specifies the name of the LDIF (LDAP Data Interchange Format) file that contains the data to be added to the LDAP server. The -f option denotes the filename

7. Create user with custom attribute

```
vim user1.ldif
```

```
dn: cn=Admins,ou=Support,dc=keenable,dc=in
objectClass: posixGroup
cn: Test
gidNumber: 4000

dn: uid=user1,ou=dev,dc=keenable,dc=in
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: user1
sn: shankar
givenName: user1
cn: user1
uidNumber: 5000
gidNumber: 5000
userPassword: redhat
loginShell: /bin/bash
homeDirectory: /home/user1
```

> In this file we have gave objectclass name and must custom attribute

a. After that add user to db

```
ldapadd -a -c -x -H ldap://localhost:3389 -D "cn=Directory Manager" -W  -f user1.ldif
```

- **f user1.ldif:** This option specifies the name of the LDIF (LDAP Data Interchange Format) file that contains the data to be added to the LDAP server. The -f option is followed by the filename (**user1.ldif** in this case).

b. **Check user reflected or not through ldapsearch command**

```
ldapsearch -x -D "cn=Directory Manager" -W -H ldap://localhost:3389 -b "ou=dev,dc=keenab
```

- **b "ou=dev,dc=keenable,dc=in"**: This option specifies the base DN (Distinguished Name) from which the search should start. It's set to **"ou=dev,dc=keenable,dc=in"** which indicates that the search should start from the "ou=dev" organizational unit under the base DN "dc=keenable,dc=in".

Output look like this

```
ravi@serverldp:~/389ds/data/ldif$ ldapsearch -x -D "cn=Directory Manager" -W -H ldap://l
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <ou=dev,dc=keenable,dc=in> with scope subtree
# filter: (uid=user1)
# requesting: ALL
#

# user1, dev, keenable.in
dn: uid=user1,ou=dev,dc=keenable,dc=in
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
objectClass: organizationalPerson
objectClass: top
objectClass: person
uid: user1
sn: shankar
givenName: user1
cn: user1
uidNumber: 5000
gidNumber: 5000
loginShell: /bin/bash
homeDirectory: /home/user1
shadowLastChange: 19600
userPassword:: e1BCS0RGMi1TSEE1MTJ9MTAwMDAkdUZoMTM3RVc0TXprQmwvRkx4dTNmQ21FVFV
 NOHRRcVkkelNzbGI0RURKUzF0YUxFVUF2eHpCZE90TDBCWjJUM25GaXhFR3BzRy9RL3JiOFJkeVpI
 b2EwME1DdW1qeTg0STZ5bExyWXpvSkxuemFRVEIyWVZmV1E9PQ==

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

8. Setup ldap Client on another VM

b. Install package related to ldap client

```
sudo apt -y install libnss-ldap libpam-ldap ldap-utils
```

- **Libnss-ldap:** It provides the necessary libraries to enable the LDAP (Lightweight Directory Access Protocol) Name Service Switch (NSS) module. This module allows us to use LDAP as a source for user, group, and other system information.

- **libpam-ldap:** It provides the necessary libraries to enable the LDAP Pluggable Authentication Module (PAM) module. This module allows us to use LDAP for user authentication.
- **ldap-utils:** This is a collection of command-line utilities for interacting with LDAP servers. These utilities are useful for managing and querying directory services.

c. After run this command we get one pop up screen

> Enter LDAP URI: IP address or hostname

> Enter Set a Distinguished name(dn) of the search base

d. Open **/etc/nslcd.conf** and check configuration

```
# /etc/nslcd.conf
# nslcd configuration file. See nslcd.conf(5)
# for details.

# The user and group nslcd should run as.
uid nslcd
gid nslcd

# The location at which the LDAP server(s) should be reachable.
uri ldap://192.168.122.109:3389

# The search base that will be used for all queries.
base dc=keenable,dc=in

# The LDAP protocol version to use.
#ldap_version 3

# The DN to bind with for normal lookups.
binddn cn=Directory Manager
bindpw ravi

# The DN used for password modifications by root.
#rootpwmoddn cn=admin,dc=example,dc=com

# SSL options
#ssl off
#tls_reqcert never
tls_cacertfile /etc/ssl/certs/ca-certificates.crt

# The search scope.
#scope sub
```

d. Restart nslcd and nscd service

```
sudo systemctl restart nslcd
```

```
sudo systemctl restart nscd
```

```
reboot
```

nslcd stands for Name Service LDAP Client Daemon. It is responsible for querying LDAP directories (such as OpenLDAP) for user and group information and providing it to local processes and services.

e. Run **getent passwd** command to check server user reflect or not

```
getent passwd user1
```

output

```
user1:x:5000:5000:user1:/home/user1:/bin/bash
```

Reference Link:-

For Understand Ldap:- https://www.windows-active-directory.com/active-directory-ldap.html

For Attribute Syntax:- https://ldap.com/attribute-syntaxes/

For Client Setup:-

https://computingforgeeks.com/how-to-configure-ubuntu-as-ldap-client/?expand_article=1&expand_article=1