# Server:

sudo su -

sudo nano /etc/hostname

#Substituim la línea per server.ldap.com

reboot

$ hostname

server.ldap.com

sudo su -

apt-get install slapd ldap-utils

s

Contraseña del administrador:

\*\*\*\*\*\*\*\*\*\*\*

nano /etc/ldap/ldap.conf

BASE dc=ldap,dc=com

URI ldap://192.168.28.116:389

dpkg-reconfigure slapd

¿Desea omitir…?

<No>

DNS

ldap.com

Nombre organización

ldap.com

contraseña

Motor de base de datos: HDB

Borre base de datos: <No>

Mover base de datos antigua: <Sí>

Protocolo ldapv2: <No>

apt-get install phpldadmin

s

nano /etc/phpldapadmin/config.php

#change those lines

$servers->setValue(‘server’,’host’,’192.168.28.116’);

$servers->setValue(‘server’,’base’,array(’dc=ldap,dc=com’));

$servers->setValue(‘login’,’bind\_id’,’cd=admin,dc=ldap,dc=com’);

// $config->custom-appearence[‘hide\_template\_warning’ = true;

systemctl restart apache2

#open mozilla firefox from the client

192.168.28.116/phpldapadmin

click:

login

dc=ldap,dc=com

create a new child entry

Generic organizational unit

Input: people

create object

ou=people

create a new child entry

Generic: Posix Group

Input: groups

create object

commit

+ou=people

cn=groups

create a new child entry

Generic: User account

GID Number:groups

Last name: raul

Login shell: /bin/sh

Password: \*\*\*\*\*\*\*\*

Create object

Açò és soles en el client:

nano /etc/share/pam-configs/mkhomedir

Name: activate mkhomedir

Default: yes

Priority: 900

Session-Type: Additional

Session:

required pam\_mkhomedir.so umask=0022 skel=/etc/skel

# Client:

apt-get install libnss-ldap libpam-ldap nscd

s

ldap://192.168.28.116

dc=ldap,dc=com

version 3

cn=admin,dc=ldap,dc=com

Password: \*\*\*\*\*\*\*\*

LDAP admin account to behave like local root?: <No>

LDAP database require login: <No>

dpkg-reconfigure libnss-ldap

ldap://192.168.28.116

dc=ldap,dc=com

3

LDAP database require login: <No>

LDAP privileges for root?: <No>

Configuration file readable/writeable: <No>

Ok

nano /etc/ldap/ldap.conf

BASE dc=ldap,dc=com

URI ldap://192.168.28.116

nano /etc/nsswitch.conf

[...]  
passwd: compat ldap  
group: compat ldap  
shadow: compat ldap  
[...]  
netgroup: ldap  
[...]

systemctl restart nscd

nano /etc/pam.d/common-auth

[...]  
auth [success=2 default=ignore] pam\_unix.so nullok\_secure try\_first\_pass  
auth [success=1 default=ignore] pam\_ldap.so use\_first\_pass  
[...]  
auth requisite pam\_deny.so  
[...]  
auth required pam\_permit.so  
[...]

nano /etc/pam.d/common-account

[...]  
account [success=2 new\_authtok\_reqd=done default=ignore] pam\_unix.so  
account [success=1 default=ignore] pam\_ldap.so  
[...]  
account requisite pam\_deny.so  
[...]  
account required pam\_permit.so  
[...]

nano /etc/pam.d/common-password

[...]  
password [success=2 default=ignore] pam\_unix.so obscure sha512  
password [success=1 user\_unknown=ignore default=die] pam\_ldap.so use\_authtok try\_first\_pass  
[...]  
password requisite pam\_deny.so  
[...]  
password required pam\_permit.so  
[...]

nano /etc/pam.d/common-session

[...]  
session required pam\_mkhomedir.so

nano /etc/pam.d/common-session-noninteractive

[...]  
session [default=1] pam\_permit.so  
[...]  
session requisite pam\_deny.so  
[...]  
session required pam\_permit.so  
[...]  
session required pam\_unix.so  
session optional pam\_ldap.so

systemctl restart nscd

Replication

## **Replication**

The LDAP service becomes increasingly important as more networked systems begin to depend on it. In such an environment, it is standard practice to build redundancy (high availability) into LDAP to prevent havoc should the LDAP server become unresponsive. This is done through LDAP replication.

Replication is achieved via the Syncrepl engine. This allows changes to be synchronized using a Consumer - Provider model. The specific kind of replication we will implement in this guide is a combination of the following modes: refreshAndPersist and delta-syncrepl. This has the Provider push changed entries to the Consumer as soon as they're made but, in addition, only actual changes will be sent, not entire entries.

### **Provider Configuration**

Begin by configuring the Provider.

Create an LDIF file with the following contents and name it provider\_sync.ldif:

# Add indexes to the frontend db.  
dn: olcDatabase={1}mdb,cn=config  
changetype: modify  
add: olcDbIndex  
olcDbIndex: entryCSN eq  
-  
add: olcDbIndex  
olcDbIndex: entryUUID eq  
  
#Load the syncprov and accesslog modules.  
dn: cn=module{0},cn=config  
changetype: modify  
add: olcModuleLoad  
olcModuleLoad: syncprov  
-  
add: olcModuleLoad  
olcModuleLoad: accesslog  
  
# Accesslog database definitions  
dn: olcDatabase={2}mdb,cn=config  
objectClass: olcDatabaseConfig  
objectClass: olcMdbConfig  
olcDatabase: {2}mdb  
olcDbDirectory: /var/lib/ldap/accesslog  
olcSuffix: cn=accesslog  
olcRootDN: cn=admin,dc=example,dc=com  
olcDbIndex: default eq  
olcDbIndex: entryCSN,objectClass,reqEnd,reqResult,reqStart  
  
# Accesslog db syncprov.  
dn: olcOverlay=syncprov,olcDatabase={2}mdb,cn=config  
changetype: add  
objectClass: olcOverlayConfig  
objectClass: olcSyncProvConfig  
olcOverlay: syncprov  
olcSpNoPresent: TRUE  
olcSpReloadHint: TRUE  
  
# syncrepl Provider for primary db  
dn: olcOverlay=syncprov,olcDatabase={1}mdb,cn=config  
changetype: add  
objectClass: olcOverlayConfig  
objectClass: olcSyncProvConfig  
olcOverlay: syncprov  
olcSpNoPresent: TRUE  
  
# accesslog overlay definitions for primary db  
dn: olcOverlay=accesslog,olcDatabase={1}mdb,cn=config  
objectClass: olcOverlayConfig  
objectClass: olcAccessLogConfig  
olcOverlay: accesslog  
olcAccessLogDB: cn=accesslog  
olcAccessLogOps: writes  
olcAccessLogSuccess: TRUE  
# scan the accesslog DB every day, and purge entries older than 7 days  
olcAccessLogPurge: 07+00:00 01+00:00

Change the rootDN in the LDIF file to match the one you have for your directory.

Create a directory:

sudo -u openldap mkdir /var/lib/ldap/accesslog

Add the new content:

sudo ldapadd -Q -Y EXTERNAL -H ldapi:/// -f provider\_sync.ldif

If you try to run this now, you will get an error:

$ ldapadd -r -Y EXTERNAL -H ldapi:/// -f /tmp/new\_database.ldif  
SASL/EXTERNAL authentication started  
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth  
SASL SSF: 0  
adding new entry "olcDatabase={2}hdb,cn=config"  
ldap\_add: Other (e.g., implementation specific) error (80)  
 additional info: olcDbDirectory: value #0: invalid path: No such file or directory

This is because the directory /var/lib/ldap-newdb/ does not exist. We can create it with:

$ sudo mkdir /var/lib/ldap-newdb/  
$ sudo chown openldap:openldap /var/lib/ldap-newdb/

There is a distinct possibility that if you try to run this now, you will get the following error:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | # ldapadd -r -Y EXTERNAL -H ldapi:/// -f /tmp/new\_database.ldif  SASL/EXTERNAL authentication started  SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth  SASL SSF: 0  adding new entry "olcDatabase={2}hdb,cn=config"  ldap\_add: Other (e.g., implementation specific) error (80)  additional info: olcDbDirectory: value #0: invalid path: Permission denied |

This is actually an ‘apparmor’ issue. App Armor is a Linux application that limits the access from certain applications to certain directories. In this case, we need to expand the permissions, which I will do without too much description, but perhaps with some luck I’ll come back to later. Modify the file /etc/apparmor.d/usr.sbin.slapd with the following lines (I list the three lines before so that context is included:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | # the databases and logs  /var/lib/ldap/ r,  /var/lib/ldap/\*\* rwk,    # The new database location  /var/lib/ldap-newdb/ r,  /var/lib/ldap-newdb/\*\* rwk, |

Run sudo service apparmor reload and try to run the file again:

root@redmine-automation:/etc/apparmor.d# ldapadd -r -Y EXTERNAL -H ldapi:/// -f /tmp/new\_database.ldif

SASL/EXTERNAL authentication started

SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth

SASL SSF: 0

adding new entry "olcDatabase={2}hdb,cn=config"

sudo -u openldap mkdir /var/lib/ldap/accesslog

The Provider is now configured.

### **Consumer Configuration**

And now configure the Consumer.

2. Install the software by going through [Instal·lació](https://help.ubuntu.com/lts/serverguide/openldap-server.html#openldap-server-installation). Make sure the slapd-config database is identical to the Provider's. In particular, make sure schemas and the databse suffix are the same.

5. Create an LDIF file with the following contents and name it consumer\_sync.ldif:
6. dn: cn=module{0},cn=config  
   changetype: modify  
   add: olcModuleLoad  
   olcModuleLoad: syncprov  
     
   dn: olcDatabase={1}mdb,cn=config  
   changetype: modify  
   add: olcDbIndex  
   olcDbIndex: entryUUID eq  
   -  
   add: olcSyncRepl  
   olcSyncRepl: rid=0 provider=ldap://ldap01.example.com bindmethod=simple binddn="cn=admin,dc=example,dc=com"  
    credentials=secret searchbase="dc=example,dc=com" logbase="cn=accesslog"  
    logfilter="(&(objectClass=auditWriteObject)(reqResult=0))" schemachecking=on  
    type=refreshAndPersist retry="60 +" syncdata=accesslog  
   -  
   add: olcUpdateRef  
   olcUpdateRef: ldap://ldap01.example.com
8. Ensure the following attributes have the correct values:
9. * provider (Provider server's hostname -- ldap01.example.com in this example -- or IP address)
   * binddn (the admin DN you're using)
   * credentials (the admin DN password you're using)
   * searchbase (the database suffix you're using)
   * olcUpdateRef (Provider server's hostname or IP address)
   * rid (Replica ID, an unique 3-digit that identifies the replica. Each consumer should have at least one rid)
11. Add the new content:
12. sudo ldapadd -Q -Y EXTERNAL -H ldapi:/// -f consumer\_sync.ldif

You're done. The two databases (suffix: dc=example,dc=com) should now be synchronizing.

### **Testing**

Once replication starts, you can monitor it by running

ldapsearch -z1 -LLLQY EXTERNAL -H ldapi:/// -s base -b dc=example,dc=com contextCSN  
  
dn: dc=example,dc=com  
contextCSN: 20120201193408.178454Z#000000#000#000000

on both the provider and the consumer. Once the output (20120201193408.178454Z#000000#000#000000 in the above example) for both machines match, you have replication. Every time a change is done in the provider, this value will change and so should the one in the consumer(s).

If your connection is slow and/or your ldap database large, it might take a while for the consumer's contextCSN match the provider's. But, you will know it is progressing since the consumer's contextCSN will be steadly increasing.

If the consumer's contextCSN is missing or does not match the provider, you should stop and figure out the issue before continuing. Try checking the slapd (syslog) and the auth log files in the provider to see if the consumer's authentication requests were successful or its requests to retrieve data (they look like a lot of ldapsearch statements) return no errors.

To test if it worked simply query, on the Consumer, the DNs in the database:

sudo ldapsearch -Q -LLL -Y EXTERNAL -H ldapi:/// -b dc=example,dc=com dn

You should see the user 'john' and the group 'miners' as well as the nodes 'People' and 'Groups'.