OpenLDAP

OpenLDAP是轻型目录访问协议（Lightweight Directory Access Protocol）的开源实现，基于X.500标准，支持TCP/IP。

它主要包括下述4个部分：

* slapd - 独立LDAP守护服务
* slurpd - 独立的LDAP更新复制守护服务
* 实现LDAP协议的库
* 工具软件和示例客户端

# 安装配置

## 安装BerkeleyDB

[root@node1 ~]# rpm -qa | grep db4 //查看Berkeley DB版本

官网上说：openLDAP不支持4.3版本，CentOS 6.4安装的是db4.7无影响，CentOS 5.8安装db4.3需删除后重新安装

Berkeley DB 4.2 (latest revision with patches) should be used with OpenLDAP 2.1 (and later). Berkeley DB 4.3 is **unstable** and should **not** be used. Versions 4.2, 4.4, and 4.5 are known to work with OpenLDAP 2.3. Versions 4.2, 4.4, 4.5, and 4.6 are known to work with OpenLDAP 2.4 and newer.

<http://www.openldap.org/faq/data/cache/44.html>

[root@node1 ~]# rpm -e --nodeps db4

[root@node1 ~]# rpm -e --nodeps db4-devel

[root@node1 ~]# ls /usr/lib | grep libdb //查看没有libdb4的库

Oracle官网注册登录，下载db-4.4.20.tar.gz，BerkeleyDB是OpenLDAP的目录数据库。

[root@node0 software]# tar zxvf db-4.4.20.tar.gz

[root@node0 software]# cd db-4.4.20/build\_unix/

[root@node0 build\_unix]# ../dist/configure

[root@node0 build\_unix]# make

[root@node0 build\_unix]# make install

[root@node0 build\_unix]# vi /etc/ld.so.conf

/usr/local/BerkeleyDB.4.4/lib

[root@node0 lib]# ldconfig

[root@node0 build\_unix]# cp /usr/local/BerkeleyDB.4.4/lib/\* /usr/lib

[root@node0 build\_unix]# cp /usr/local/BerkeleyDB.4.4/include/\* /usr/include

//由于把系统自带的db删掉，会造成httpd启动出错

[root@node0 build\_unix]# cd /usr/lib

[root@node0 lib]# ln –s libdb-4.4.so libdb-4.3.so

## 安装OpenLDAP

[root@node0 ~]# service iptables stop //或增加389端口

[root@node0 ~]# chkconfig iptables off

[root@node0 ~]# vi /etc/selinux/config 或setenforce 0

SELINUX=disabled

1. **Yum安装**

[root@node0 ~]# yum -y install openldap openldap-servers openldap-clients

[root@node0 ~]# yum -y install sssd perl-LDAP.noarch

1. **源码安装**

依赖软件：

* openSSL：提供交互图层安全服务；

[root@node0 openldap-1.2.9]# yum install openssl openssl-devel

* Cyrus SASL：提供简单认证和安全认证服务；须先安装openSSL。

[root@node0 openldap-1.2.9]# yum install cyrus-sasl cyrus-sasl-devel cyrus-sasl-gssapi

* Kerberos：OpenLDAP支持认证机制，如果想提供强健的认证服务，推荐使用它

cyrus-sasl-md5 cyrus-sasl-plain

[root@node0 openldap-1.2.9]# yum install krb5-server krb5-libs krb5-workstation

* BDB/HDB：OpenLDAP的数据库

[root@node0 openldap-1.2.9]# yum install db4 db4-devel

[root@node0 software]# tar zxvf openldap-2.4.35.tgz

[root@node0 software]# cd openldap-2.4.35

[root@node0 openldap-2.4.35]# ./configure –help

//使用非系统的Berkeley DB

[root@node0 openldap-2.4.35]# env CPPFLAGS="-I/usr/local/BerkeleyDB.4.4/include" LDFLAGS="-L/usr/local/BerkeleyDB.4.4/lib" ./configure --prefix=/usr/local/openldap

//使用系统自带的Berkeley DB

[root@node0 openldap-2.4.35]# ./configure --prefix=/usr/local/openldap

**问题1：configure: error: BerkeleyDB version incompatible with BDB/HDB backends**

原因1：检测到的BerkeleyDB为4.3版本，会出现major=4,minor=3提示

原因2：系统自带的Berkeley DB与安装的DB混淆，可通过CPPFLAGS，LDFLAGS指定DB，并将BerkeyleyDB的include，lib目录内容拷贝到/usr/include和/usr/lib中

**问题2：找不到ltdl.h文件，error: could not locate libtool ltdl.h**

[root@node0 include]# locate ltdl.h

ltdl.h在libltdl3-devel包中

[root@node0 software]# rpm -ivh libtool-ltdl-devel-1.5.22-7.el5\_4.i386.rpm

或[root@node0 software]# yum install libtool-ltdl-devel

**问题3：configure: error: select appropriate LDBM options or disable**

找不到gdbm的头文件

[root@node0 openldap-1.2.9]# yum install gdbm gdbm-devel

//检测成功后

[root@node0 openldap-2.4.35]# make depend

[root@node0 openldap-2.4.35]# make

[root@node0 openldap-2.4.35]# make test

[root@node0 openldap-2.4.35]# make install

[root@node1 ~]# vi ~/.bashrc

PATH=$PATH:/usr/local/openldap/bin

LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/openldap/lib

export PATH

export LD\_LIBRARY\_PATH

[root@node1 ~]# source ~/.bashrc

## 配置服务器端

[root@node0 ~]# updatedb //更新locate数据库

//slapd.conf在/usr/share/openldap-servers下

[root@node0 ~]# cp /usr/share/openldap-servers/slapd.conf.obsolete /etc/openldap/slapd.conf

[root@node0 ~]# slappasswd

{SSHA}niDBT9UkakrsZlS3qyH5lydAfv3lMqet

[root@node0 ~]# vi /etc/openldap/slapd.conf

by dn.exact="cn=Manager,dc=scs,dc=com" read

database bdb

suffix "dc=scs,dc=com"

checkpoint 1024 15

rootdn "cn=Manager,dc=scs,dc=com"

rootpw {SSHA}niDBT9UkakrsZlS3qyH5lydAfv3lMqet

// ldap.conf在/etc/openldap下

[root@node0 ~]# cp /etc/openldap/ldap.conf /etc/openldap/ldap.conf.bak

[root@node0 ~]# vi /etc/openldap/ldap.conf

URI ldap://127.0.0.1

BASE dc=scs,dc=com

TLS\_CACERTDIR /etc/openldap/certs

[root@node0 ~]# vi /etc/sysconfig/ldap

SLAPD\_LDAPI=yes

[root@node0 ~]# cp /usr/share/openldap-servers/DB\_CONFIG.example /var/lib/ldap/DB\_CONFIG

//由于slapd以ldap用户运行，如果目录数据库的所有者不是ldap的话，slapd就无法写入数据

[root@node0 openldap]# chown -R ldap.ldap /var/lib/ldap

[root@node0 ~]# ls -ld /var/lib/ldap

drwx------ 2 ldap ldap 4096 04-07 11:20 /var/lib/ldap

//验证配置文件是否正确并生成slapd.d目录

[root@node0 openldap]# rm -rf /etc/openldap/slapd.d/\*

[root@node0 openldap]# slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d

config file testing succeeded

[root@node0 openldap]# chown -R ldap:ldap /etc/openldap/slapd.d

## 启动slapd

YUM安装:

[root@node0 ~]# service slapd start

源码安装：

[root@node0 ~]# /usr/local/openldap/libexec/slapd //将其加入PATH

[root@node0 ~]# chkconfig --levels 235 slapd on

检查SLAPD是否运行：

[root@node0 ~]# ps aux | grep slapd

[root@node0 ~]# netstat -tunlp | grep 389

[root@node0 ~]# ldapsearch -x -b '' -s base '(objectclass=\*)' namingContexts

**问题1: ldap\_bind: Can't contact LDAP server (-1)**

原因是slapd没有启动

/usr/local/openldap/libexec/slapd -d 254 查看启动失败的原因

<http://bbs.chinaunix.net/forum.php?mod=viewthread&tid=717756>

**问题2：make test出错**

系统自带的db4与安装的Berkley DB冲突

解决方法：保证系统中只有一个DB版本

[root@node0 lib]# yum search db4

DB与openLDAP的版本不符

**问题3：ldap\_bind: Invalid credentials (49)**

可能是参数错误，例如-D "cn=Manager,dc=scs,dc=cn"应该为com

**问题4：ldap\_bind: Invalid DN syntax (34) additional info: invalid DN**

[root@node0 openldap]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f /home/dwt/software/base.ldif

-D后面的内容不对，例如引号是否英文

**问题5：locate: can not stat () `/var/lib/mlocate/mlocate.db': No such file or directory**

[root@node0 openldap]# updatedb

**问题6：ln: accessing `/var/run/openldap/slapd.pid': No such file or directory**

[root@node0 openldap]# touch /var/run/openldap/slapd.pid

**问题7：bdb\_db\_open: database unclean shutdown detected; attempting recovery**.

非正常关机导致db出错

[root@node0 openldap]# db\_recover

或重建数据库

## 数据迁移

参考<http://www.padl.com/OSS/MigrationTools.html>

[root@node0 ~]# yum -y install migrationtools

[root@node0 ~]# cd /usr/share/migrationtools

//修改整体配置

[root@node0 migrationtools]# vi migrate\_common.ph

# Default DNS domain

$DEFAULT\_MAIL\_DOMAIN = "scs.com";

# Default base

$DEFAULT\_BASE = "dc=scs,dc=com";

[root@node0 migrationtools]# ./migrate\_base.pl > base.ldif //运行migrate\_base.pl，生成base.ldif，base.ldif创建Host,Networks,Group,People等一系列组织单元，即表基本结构，删除到只剩People和Group

[root@node0 migrationtools]# vi base.ldif

dn: dc=scs,dc=com

dc: scs

objectClass: top

objectClass: domain

dn: ou=People,dc=scs,dc=com

ou: People

objectClass: top

objectClass: organizationalUnit

dn: ou=Group,dc=scs,dc=com

ou: Group

objectClass: top

objectClass: organizationalUnit

[root@node0 migrationtools]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f base.ldif

**方法1：导入文件中的某个用户**

//新增用户

[root@node0 migrationtools]# useradd ldapuser

[root@node0 migrationtools]# passwd ldapuser

[root@node0 migrationtools]# cat /etc/passwd | grep ldapuser > passwd.in

[root@node0 migrationtools]# ./migrate\_passwd.pl passwd.in > passwd.ldif

[root@node0 migrationtools]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f passwd.ldif

[root@node0 migrationtools]# cat /etc/group | grep ldapuser > group.in

[root@node0 migrationtools]# ./migrate\_group.pl group.in > group.ldif

[root@node0 migrationtools]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f group.ldif

//删除该用户

直接删除/etc/passwd和/etc/group中的ldapuser行

**方法2：导入文件**

//将/etc/group转变为ldif格式，再导入ldap数据库

[root@node0 migrationtools]# ./migrate\_group.pl /etc/group > group.ldif

[root@node0 migrationtools]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f group.ldif

[root@node0 migrationtools]# ./migrate\_passwd.pl /etc/passwd > passwd.ldif

[root@node0 migrationtools]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f passwd.ldif

**方法3：用shell脚本导入全部**

[root@node0 migrationtools]# ./migrate\_all\_online.sh

Enter the X.500 naming context you wish to import into: [dc=scs,dc=com] //回车

Enter the hostname of your LDAP server [ldap]: localhost //输入LDAP的hostname

Enter the manager DN: [cn=manager,dc=scs,dc=com]: //回车

Enter the credentials to bind with: //输入目录管理员manager的密码

Do you wish to generate a DUAConfigProfile [yes|no]? //回车

**问题1：/etc/hosts和/etc/services导入会出错**

/etc/hosts下有

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

会导致加入2个localhost而报错，将执行语句关闭或将这两行注释掉

/etc/services的nextstep，raid-am导入会出错，将执行语句关闭或nextstep两行注释#掉

每个shell脚本的适用情况如下：

|  |  |  |
| --- | --- | --- |
| Shell script | Existing nameservice | LDAP running? |
| migrate\_all\_online.sh | /etc flat files | Yes |
| migrate\_all\_offline.sh | /etc flat files | No |
| migrate\_all\_netinfo\_online.sh | NetInfo | Yes |
| migrate\_all\_netinfo\_offline.sh | NetInfo | No |
| migrate\_all\_nis\_online.sh | NIS/YP | Yes |
| migrate\_all\_nis\_offline.sh | NIS/YP | No |

//查看结果

[root@node0 migrationtools]# ldapsearch -x

## 配置网页

//先安装yum repo

[root@node0 ~]# rpm -Uvh <http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm>

[root@node0 ~]# ls /etc/yum.repos.d/

epel.repo epel-testing.repo

[root@node0 ~]# yum -y install phpMyAdmin

[root@node0 ~]# vi /etc/httpd/conf.d/phpMyAdmin.conf

<Directory /usr/share/phpMyAdmin/>

<IfModule !mod\_authz\_core.c>

# Apache 2.2

Order Deny,Allow

Deny from All

Allow from 127.0.0.1 192.168.0.0/24

Allow from ::1

</IfModule>

</Directory>

[root@node0 ~]# yum -y install phpldapadmin

[root@node0 ~]# vi /etc/httpd/conf.d/phpldapadmin.conf

<Directory /usr/share/phpldapadmin/htdocs>

Order Deny,Allow

Deny from all

Allow from 127.0.0.1 192.168.0.0/24

Allow from ::1

</Directory>

[root@node0 ~]# vi /etc/phpldapadmin/config.php

$config->custom->appearance['hide\_template\_warning'] = true; //将注释去掉，false

$servers->setValue('login','attr','dn'); //这行注释去掉

// $servers->setValue('login','attr','uid'); //这行注释起来

[root@node0 ~]# service httpd restart

[root@node0 ~]# chkconfig httpd on

登陆<http://192.168.0.59/phpldapadmin/>

DN输入cn=Manager,dc=scs,dc=com，密码123456

**问题1：Could not reliably determine the server's fully qualified domain name, using ServerName**

[root@node0 ~]# vi /etc/httpd/conf/httpd.conf

#ServerName www.example.com:80

ServerName localhost

[root@node0 ~]# service httpd restart

## TLS连接

### 生成公私钥

LDAP 是以明文的格式通过网络来发送所有信息的，包括密码。TLS 提供加密机制（SSL 的后继者）来解决这个问题。在传输层，数据使用 TLS 协议进行加密和封装，然后通过网络进行传输。OpenSSL 包提供配置加密的工具。

* openssl：包括一个证书管理工具和提供各种加密算法和协议的共享库

[root@node0 ~]# cd /etc/pki/tls/misc

1. **创建主证书**

PEM（Privacy Enhanced Mail，私密性增强邮件）是一种用来对数据进行加密和编码的格式

[root@node0 misc]# ./CA -newca //如果出错，则rm -fr /etc/pki/CA

CA certificate filename (or enter to create) //回车

Enter PEM pass phrase: //输入密码

Verifying - Enter PEM pass phrase: //重复密码

Country Name (2 letter code) [AU]:cn

State or Province Name (full name) [Some-State]:cn

Locality Name (eg, city) []:cn

Organization Name (eg, company) [Internet Widgits Pty Ltd]:cn

Organizational Unit Name (eg, section) []:cn

Common Name (eg, YOUR name) []:node0   //域名

Email Address []: //回车

Enter pass phrase for /etc/pki/CA/private/./cakey.pem: //再次输入以上密码

1. **生成CA签名的服务器认证**

-nodes设置不用每次启动都输入密码，签名的公钥保存在证书请求slapd-req.pem中，私钥保存在slapd-key.pem中

[root@node0 misc]# openssl req -new -nodes -keyout slapd-key.pem -out newreq.pem -days 365

1. **利用CA认证公钥**

[root@node0 misc]# ./CA –sign

[root@node0 ~]# cd /etc/pki/tls/certs

[root@node0 certs]# make server.key

[root@node0 certs]# openssl rsa -in server.key -out server.key

[root@node0 certs]# make server.csr

[root@node0 certs]#openssl x509 -in server.csr -out server.crt -req -signkey server.key -days 3650

[root@node0 certs]# chmod 400 server.\*

### 配置服务器端

1. **将认证和私钥拷贝到openldap认证目录下,并设置好权限**

[root@node0 misc]# cp slapd-key.pem /etc/openldap/certs/

[root@node0 misc]# cp newcert.pem /etc/openldap/certs/slapd-cert.pem

[root@node0 misc]# chown -R ldap.ldap /etc/openldap/certs

[root@node0 misc]# chmod -R 644 /etc/openldap/certs

1. **配置slapd.conf,ldap.conf**

TLSCACertificatePath 指明公钥存放路径，TLSCertificateFile TLSCertificateKeyFile指明认证和私钥的文件路径。

[root@node0 certs]# vi /etc/openldap/slapd.conf

#TLSCACertificatePath /etc/openldap/certs

#TLSCertificateFile "\"OpenLDAP Server\""

#TLSCertificateKeyFile /etc/openldap/certs/password

TLSCACertificatePath /etc/openldap/certs

TLSCertificateFile /etc/openldap/certs/slapd-cert.pem

TLSCertificateKeyFile /etc/openldap/certs/slapd-key.pem

[root@node0 certs]# vi /etc/openldap/ldap.conf

URI ldap://localhost/

BASE dc=scs,dc=com

TLS\_CACERT /etc/openldap/certs/slapd-cert.pem

TLS\_CACERTDIR /etc/openldap/cacerts

[root@node0 certs]# vi /etc/sysconfig/ldap

SLAPD\_LDAPS=yes

1. **验证TLS是否成功**

通过ldaps:协议连接，-x是指采用简单认证

[root@node0 ~]# ldapsearch -x -H ldaps://node0

如果出错，则用[root@node0 ~]# ldapsearch -d 1 -v -x -H ldaps://node0查看原因

**问题1：ldap\_sasl\_bind(SIMPLE): Can't contact LDAP server (-1)**

命令行没有加-x，则采用SASL认证，如果设置-x即SASL认证出错，则通过以下方式查看具体原因 [root@node0 certs]# ldapsearch -d 1 -v -H ldaps://localhost:636

原因可能是slapd默认启动在389端口上,并没有监听636端口，也有可能TLS/SSL握手无法建立，即TLS没有配置好

[root@node0 certs]# netstat -tunlp | grep 636

//让slapd启动在636端口上

[root@node0 certs]# vi /etc/sysconfig/ldap

SLAPD\_LDAPS=yes

[root@node0 certs]# service slapd restart

**问题2：ldap\_start\_tls: Operations error (1) TLS already started**

Ldapsearch时如果同时使用-ZZ和-H ldaps://localhost，或者ldap.conf的URI中设置ldaps://localhost则会报此错。

**问题3：Credentials cache file '/tmp/krb5cc\_0' not found**

因为查询的主机名不存在于~/.ssh/known\_hosts，用ssh+主机名将其加入known\_hosts

**问题4：TLS error -8179:Peer's Certificate issuer is not recognized.**

TLS没有配置好，公钥位置找不到

### 配置客户端

[root@node1 openldap]# yum install openldap openldap-clients

//将服务器的公钥发送到客户端上

[root@node0 openldap]# scp /etc/openldap/certs/slapd-cert.pem node1:/etc/openldap/certs

//让ldapsearch，ldapadd能够找到公钥，需设置TLS\_CACERT的公钥路径

[root@node1 openldap]# vi /etc/openldap/ldap.conf

TLS\_CACERT /etc/openldap/certs/slapd-cert.pem

URI ldap://node0/

BASE dc=scs,dc=com

ssl start\_tls

TLS\_CACERTDIR /etc/openldap/cacerts

//检验TLS是否设置成功

[root@node1~]# ldapsearch -h node0 -x -b "dc=scs,dc=com" –ZZ

//以明文方式查询服务器

[root@node1 ~]# ldapsearch -x -b "dc=scs,dc=com" -H ldap://node0

//以TLS方式查询服务器

[root@node1 ~]# ldapsearch -x -b "dc=scs,dc=com" -H ldaps://node0

[root@node1 ~]# openssl s\_client -connect node0:636

## LDAP登录

LDAP 身份验证需要配置两个服务：系统命名服务和身份验证服务。

* nss\_ldap 是一组 C 库扩展，它允许 LDAP 目录服务器用作一个用户和组信息的主源
* pam\_ldap 是一个 Linux-PAM 模块，它支持身份验证功能

系统命名服务（NSS）需要配置为使用 LDAP 来解析诸如用户和组帐号之类的资源。例如，在运行命令 ls -l 时，如果某个文件 inode 给出文件的所有者是 “user 501”，那么命名服务就需要将 “uid 501” 解析成用户名，并在 ls 命令输出结果中输出。通常来说，这是通过查找 /etc/passwd 文件中的所有用户帐号实现的。由于用户现在都存储在 LDAP 目录中，因此系统需要配置成同时对 passwd 文件和 LDAP 目录中的帐号进行解析。这种功能是通过 /usr/lib/libnss\_ldap.so 库提供的。

身份验证服务是向 LDAP 验证用户身份的服务。可插入身份验证模块（PAM）提供了本地 Linux 身份验证服务。首先配置 PAM 先对本地的 /etc/passwd 文件检查用户帐号，然后再对 LDAP 服务器进行检查。PAM LDAP 模块可以用来将身份验证重定向到 LDAP 目录上。/lib/security/pam\_ldap.so PAM 模块提供了 LDAP 身份验证功能。

身份验证本身是由 PAM 程序执行的，它从身份验证候选机制中获取用户名，将其绑定到 OpenLDAP 服务器上，检索与这个 uid 条目（用户名条目）相关的 DN；从身份验证候选机制中获取密码，然后使用这个 DN 和密码试图将其绑定到 OpenLDAP 服务器上。如果绑定成功，PAM 会报告说这个用户已经成功通过了 pam\_ldap.so 提供的身份验证测试。根据 PAM 的配置不同，在用户看到命令行提示符之前可能会执行其他测试。

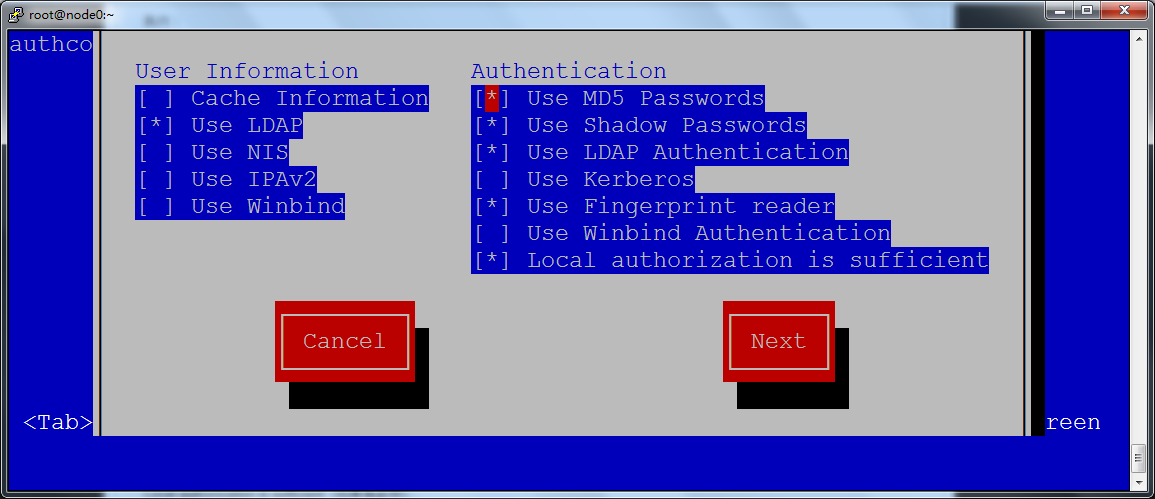
首先安装pam\_ldap，nss：

[root@node1 openldap]# yum install nss-pam-ldapd

然后可采用两种方法来配置 LDAP 客户机。一种快速而简单的方法是运行 /usr/sbin/authconfig，输入信息。另外一种方法是通过编辑客户机 LDAP 配置文件 /etc/ldap.conf，和/etc/nsswitch.conf、/etc/sysconfig/authconfig 和 /etc/pam.d/system-auth。

**方法1：**

[root@node1 ~]# setup 或 [root@node1~]# authconfig-tui



选择

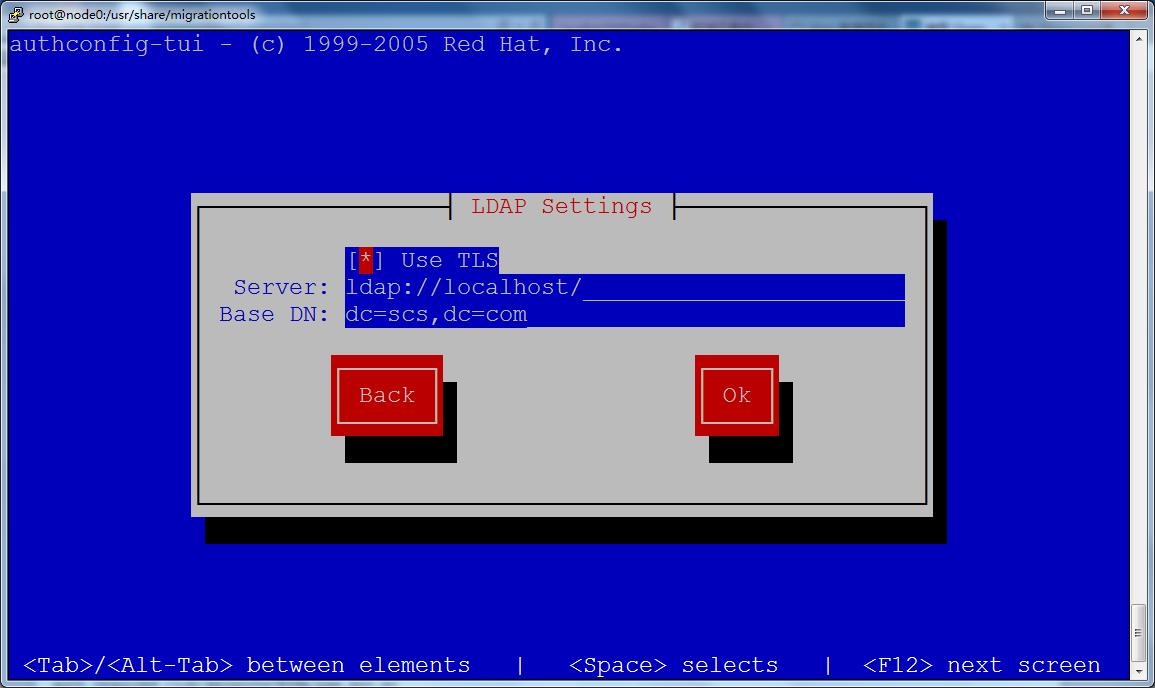
Use LDAP

Use MD5 Passwords

Use Shadow Passwords

Use LDAP Authentication

Local authorization is sufficient



如果采用TLS传输，首先必须保证TLS配置成功，否则会id ldapuser后失败

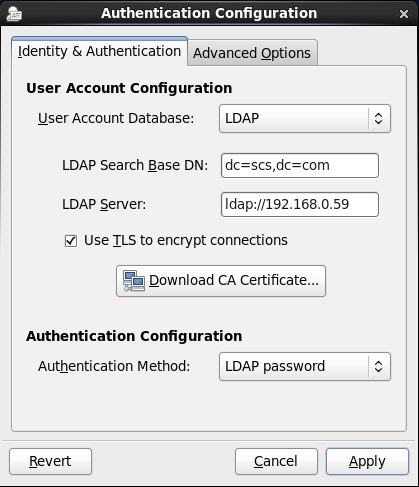
客户端选择不用TLS，服务器端选择用TLS

Server：node0

Base DN: dc=scs,dc=com

登录终端，System->Administration->Authentication

或者直接命令行运行[root@node1 ~]# system-config-authentication



**方法2：**

[root@node1 ~]# vi /etc/sysconfig/authconfig

USELDAP=yes

USELDAPAUTH=yes

USESHADOW=yes

USELOCAUTHORIZE=yes

[root@node1 ~]# vi /etc/openldap/ldap.conf

URI ldap://localhost/

BASE dc=scs,dc=com

TLS\_CACERT /etc/openldap/certs/slapd-cert.pem

TLS\_CACERTDIR /etc/openldap/cacerts

[root@node1 ~]# vi /etc/nsswitch.conf

//LDAP服务器上为sss，客户端为ldap

passwd: files ldap

shadow: files ldap

group: files ldap

netgroup: files ldap

automount: files ldap

aliases: files nisplus

[root@node1 ~]# vi /etc/nslcd.conf

//服务器端没有该文件

//客户端采用tls

uri ldap://192.168.0.59

base dc=scs,dc=com

ssl start\_tls

tls\_reqcert allow

tls\_cacertdir /etc/openldap/certs

//客户端不采用tls

uri ldap://node0/

base dc=scs,dc=com

ssl no

tls\_cacertdir /etc/openldap/cacerts

[root@node1 ~]# vi /etc/pam\_ldap.conf

//服务器端没有该文件

//客户端采用tls

uri ldap://192.168.0.59

ssl start\_tls

tls\_reqcert allow

tls\_cacertdir /etc/openldap/cacerts

pam\_password md5

//客户端不采用tls

uri ldap://node0/

ssl no

tls\_cacertdir /etc/openldap/cacerts

pam\_password md5

[root@node1 ~]# vi /etc/pam.d/system-auth

//服务器端没有pam\_ldap.so

auth required pam\_env.so

auth sufficient pam\_fprintd.so

auth sufficient pam\_unix.so nullok try\_first\_pass

auth requisite pam\_succeed\_if.so uid >= 500 quiet

auth sufficient pam\_ldap.so use\_first\_pass

auth required pam\_deny.so

account required pam\_unix.so broken\_shadow

account sufficient pam\_localuser.so

account sufficient pam\_succeed\_if.so uid < 500 quiet

account [default=bad success=ok user\_unknown=ignore] pam\_ldap.so

account required pam\_permit.so

password requisite pam\_cracklib.so try\_first\_pass retry=3 type=

password sufficient pam\_unix.so md5 shadow nullok try\_first\_pass use\_authtok

password sufficient pam\_ldap.so use\_authtok

password required pam\_deny.so

account [default=bad success=ok user\_unknown=ignore] pam\_ldap.so

session optional pam\_keyinit.so revoke

session required pam\_limits.so

session [success=1 default=ignore] pam\_succeed\_if.so service in crond quiet use\_uid

session required pam\_unix.so

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

session optional pam\_ldap.so

[root@node1 ~]# service nslcd restart

**用LDAP存入而用户ldapuser，验证是否成功：**

[root @node1 ~]$ id ldapuser

uid=501(ldapuser) gid=500(dwt) groups=500(dwt)

## 重配slapd.conf

/etc/openldap/slapd.d是OpenLDAP新采用的配置方式，重配slapd.conf后，须将slapd.d目录内容删除，重启才生效

[root@node0 openldap]# rm -rf /etc/openldap/slapd.d/\*

//利用slaptest将老的slapd.conf转换成新的slapd.d

[root@node0 openldap]# slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d

[root@node0 openldap]# chown -R ldap.ldap /etc/openldap/

[root@node0 openldap]# service slapd restart

## 数据库重建

[root@node0 openldap]# rm -rf /var/lib/ldap/\*

[root@node0 openldap]# cp /usr/share/openldap-servers/DB\_CONFIG.example /var/lib/ldap/DB\_CONFIG

[root@node0 openldap]# chown -R ldap.ldap /var/lib/ldap/

[root@node0 openldap]# service slapd restart

## 新增记录

[root@node0 openldap]# vi /home/dwt/software/base.ldif

dn: dc=scs,dc=com

dc: scs

objectClass: top

objectClass: domain

dn: ou=People,dc=scs,dc=com

ou: People

objectClass: top

objectClass: organizationalUnit

dn: ou=Group,dc=scs,dc=com

ou: Group

objectClass: top

objectClass: organizationalUnit

[root@node0 openldap]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f /home/dwt/software/base.ldif

[root@node0 openldap]# vi /home/dwt/software/group.ldif

dn: cn=demo,ou=Group,dc=scs,dc=com

objectClass: posixGroup

objectClass: top

cn: demo

userPassword: password

gidNumber: 1000

[root@node0 openldap]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f /home/dwt/software/group.ldif

[root@node0 openldap]# vi /home/dwt/software/user.ldif

dn: uid=demo,ou=People,dc=scs,dc=com

uid: demo

cn: demo user

objectClass: account

objectClass: posixAccount

objectClass: top

objectClass: shadowAccount

userPassword: password

shadowLastChange: 15140

shadowMin: 0

shadowMax: 99999

shadowWarning: 7

loginShell: /bin/bash

uidNumber: 1000

gidNumber: 1000

homeDirectory: /home/demo

[root@node0 openldap]# ldapadd -x -W -D "cn=Manager,dc=scs,dc=com" -f /home/dwt/software/user.ldif

## SASL认证（未完成）

Ldapsearch和ldapmodify默认利用SASL验证用户，SASL（Simple Authentication and Security Layer）提供数据在客户端和服务器间安全认证协议

//查看支持的SASL机制

[root@node0 sasl2]# ldapsearch -h localhost -p 389 -x -b "" -s base -LLL supportedSASLMechanisms

dn:

supportedSASLMechanisms: GSSAPI

supportedSASLMechanisms: DIGEST-MD5

supportedSASLMechanisms: CRAM-MD5

//安装SASL

[root@node0 openldap-1.2.9]# yum install cyrus-sasl cyrus-sasl-devel cyrus-sasl-gssapi nss\_ldap pam\_krb5 krb5-libs krb5-workstation

//建立Kerberos Servers

[root@node0 sasl2]# sasldblistusers2 //查看完整域名，即node0

admin@node0: userPassword

[root@node0 sasl2]# cp /etc/krb5.conf /etc/krb5.conf.bak

[root@node0 sasl2]# vi /etc/krb5.conf

[libdefaults]

default\_realm = node0

dns\_lookup\_realm = false

dns\_lookup\_kdc = false

ticket\_lifetime = 24h

renew\_lifetime = 7d

forwardable = true

[realms]

node0 = {

kdc = node0

admin\_server = node0

}

[domain\_realm]

.node0 = node0

node0 = node0

[root@node0 sasl2]# cp /var/kerberos/krb5kdc/kdc.conf /var/kerberos/krb5kdc/kdc.conf.bak

[root@node0 sasl2]# vi /var/kerberos/krb5kdc/kdc.conf

[realms]

node0 = {

#master\_key\_type = aes256-cts

acl\_file = /var/kerberos/krb5kdc/kadm5.acl

dict\_file = /usr/share/dict/words

admin\_keytab = /var/kerberos/krb5kdc/kadm5.keytab

master\_key\_type = des3-hmac-sha1

supported\_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal

}

[root@node0 sasl2]# kdb5\_util create -s //建立Kerberos数据库

[root@node0 migrationtools]# vi /var/kerberos/krb5kdc/kadm5.acl

[root@node0 sasl2]# saslpasswd2 -c admin //建立管理员，密码与OpenLDAP的相同

[root@node0 sasl2]# vi /etc/openldap/slapd.conf

//在最后增加一行

sasl-regexp uid=(.\*),cn=node0,cn=DIGEST-MD5,cn=auth uid=$1,ou=People,dc=scs,dc=com

# 使用手册

OpenLDAP包含3部分：

* openldap:包含运行OpenLDAP server和client的库
* openldap-clients:包含用于修改查看LDAP server的工具
* openldap-servers:包含配置运行LDAP server的工具，openldap-servers包括2个程序Standalone LDAP Daemon(/usr/sbin/slapd，LDAP server服务进程)和Standalone LDAP Update Replication Daemon(/usr/sbin/slurpd，同步多个LDAP server数据)。

OpenLDAP守护进程：

* slapd：主 LDAP 服务器
* slurpd：负责与复制 LDAP 服务器保持同步的服务器

对网络上的目录进行操作的客户机程序：

* ldapadd：打开一个到 LDAP 服务器的连接，绑定、修改或增加条目
* ldapsearch：打开一个到 LDAP 服务器的连接，绑定并使用指定的参数进行搜索

对本地系统上的数据库进行操作的程序：

* slapadd：将以 LDAP 目录交换格式（LDIF）指定的条目添加到 LDAP 数据库中
* slapcat：打开 LDAP 数据库，并将对应的条目输出为 LDIF 格式

配置文件都在/etc/openldap下：

* /etc/openldap/ldap.conf：客户端配置文件，ldapsearch，ldapadd等使用
* /etc/openldap/slapd.conf：slapd配置文件
* /etc/openldap/schema/：/etc/openldap/slapd.conf中指明该目录下用到的schema，新增记录必须符合schema要求

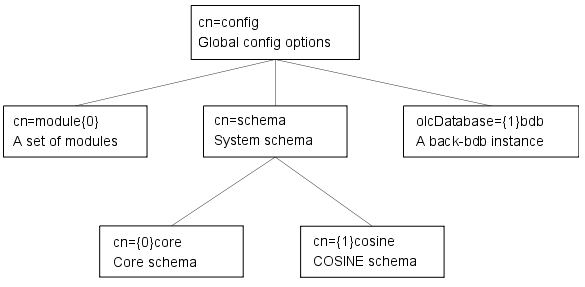
dn：distinguished name

cn：commonName

olc：OpenLDAP Configuration

## 配置文件

Slapd配置文件以LDAP目录树形式保存，顶层为cn=config，其下包含全局配置。



### slapd.d

1. # example config file - global configuration entry //定义了2个BDB数据库实例，处理X.500树的不同部分

2. dn: cn=config

3. objectClass: olcGlobal

4. cn: config

5. olcReferral: ldap://root.openldap.org //失败转移URI

6.

7. # internal schema

8. dn: cn=schema,cn=config //定义schema子树

9. objectClass: olcSchemaConfig

10. cn: schema

11.

12. # include the core schema

13. include: [file:///usr/local/etc/openldap/schema/core.ldif //](file:///usr/local/etc/openldap/schema/core.ldif%20//)包含core的schema文件

14.

15. # global database parameters //第一个数据库是frontend，其设置应用于全局

16. dn: olcDatabase=frontend,cn=config

17. objectClass: olcDatabaseConfig

18. olcDatabase: frontend

19. olcAccess: to \* by \* read //通用访问控制，应用到全部实体

20.

21. # set a rootpw for the config database so we can bind. //第二个数据库config

22. # deny access to everyone else.

23. dn: olcDatabase=config,cn=config

24. objectClass: olcDatabaseConfig

25. olcDatabase: config

26. olcRootPW: {SSHA}XKYnrjvGT3wZFQrDD5040US592LxsdLy

27. olcAccess: to \* by \* none //取消所有访问，只有超级用户可访问

28.

29. # BDB definition for example.com //第三个数据库bdb，处理dc=example,dc=com子树的查询

30. dn: olcDatabase=bdb,cn=config

31. objectClass: olcDatabaseConfig

32. objectClass: olcBdbConfig

33. olcDatabase: bdb

34. olcSuffix: dc=example,dc=com //查询后缀

35. olcDbDirectory: /usr/local/var/openldap-data //数据存放目录

36. olcRootDN: cn=Manager,dc=example,dc=com //定义超级用户和密码

37. olcRootPW: secret

38. olcDbIndex: uid pres,eq //各种属性的索引

39. olcDbIndex: cn,sn pres,eq,approx,sub

40. olcDbIndex: objectClass eq

41. olcAccess: to attrs=userPassword //密码能被用户自己和admin修改，能够用于认证/授权，但不能读

42. by self write

43. by anonymous auth

44. by dn.base="cn=Admin,dc=example,dc=com" write

45. by \* none

46. olcAccess: to \* //其他属性能够被用户自己和admin写，被所有用户读

47. by self write

48. by dn.base="cn=Admin,dc=example,dc=com" write

49. by \* read

50.

51. # BDB definition for example.net //定义另一个dbd，处理dc=example,dc=net子树

52. dn: olcDatabase=bdb,cn=config

53. objectClass: olcDatabaseConfig

54. objectClass: olcBdbConfig

55. olcDatabase: bdb

56. olcSuffix: "dc=example,dc=net"

57. olcDbDirectory: /usr/local/var/openldap-data-net

58. olcRootDN: "cn=Manager,dc=example,dc=com"

59. olcDbIndex: objectClass eq

60. olcAccess: to \* by users read

### slapd.conf

slapd.conf包含全局，后端，数据库3部分配置。

1. # example config file - global configuration section

2. include /usr/local/etc/schema/core.schema //schema位置

3. referral ldap://root.openldap.org //当查询无法连接到database时，转而查root.openldap.org的389端口

4. access to \* by \* read //通用访问控制，所有人都可访问任何位置

5. # BDB definition for the example.com

6. database bdb

7. suffix "dc=example,dc=com" //查询传到数据库中后缀为example.com的目录

8. directory /usr/local/var/openldap-data //数据库文件存放位置，所属用户必须为ldap

9. rootdn "cn=Manager,dc=example,dc=com" //超级用户记录，不受access，size，time limit限制

10. rootpw secret //超级用户密码

11. # replication directives

12. replogfile /usr/local/var/openldap/slapd.replog

13. replica uri=ldap://slave1.example.com:389

14. binddn="cn=Replicator,dc=example,dc=com"

15. bindmethod=simple credentials=secret

16. replica uri=ldaps://slave2.example.com:636

17. binddn="cn=Replicator,dc=example,dc=com"

18. bindmethod=simple credentials=secret

19. # indexed attribute definitions //保持各种属性的索引

20. index uid pres,eq

21. index cn,sn,uid pres,eq,sub

22. index objectClass eq

23. # database access control definitions //数据库访问控制

24. access to attr=userPassword //属性userPassword能被自身和Admin所写

25. by self write

26. by anonymous auth

27. by dn.base="cn=Admin,dc=example,dc=com" write

28. by \* none

29. access to \*

30. by self write

31. by dn.base="cn=Admin,dc=example,dc=com" write

32. by \* read

33. # BDB definition for example.net //又定义了一个BDB

34. database bdb

35. suffix "dc=example,dc=net"

36. directory /usr/local/var/openldap-data-net

37. rootdn "cn=Manager,dc=example,dc=com"

38. index objectClass eq

39. access to \* by users read

include /etc/openldap/schema/core.schema

include /etc/openldap/schema/cosine.schema

include /etc/openldap/schema/inetorgperson.schema

include /etc/openldap/schema/nis.schema

# 允许 LDAPv2 客户端连接

allow bind\_v2

# pidfile保存slapd的进程ID，argsfile保存slapd命令行参数

pidfile /var/run/openldap/slapd.pid

argsfile /var/run/openldap/slapd.args

# loglevel默认没有包含在配置文件中，格式为loglevel+整数或关键字，可用整数之和，默认值为256

# Logging

# - trace function calls (1)

# - connection management (8)

# - ACL processing (128)

# - stats log connections/operations/results (256)

# (1 + 8 + 128 + 256)=393

loglevel 393

logfile /var/log/ldap.log

# 如果指明loglevel，则必须在slapd启动前生成log文件

[root@node0 ~]# touch /var/log/ldap.log

# password-hash默认没有包含在配置文件中，用于配置多个哈希值，保存在userPassword属性中，以供用户生成密码时使用。可选项包括SSHA，SHA，SMD5，MD5，CRYPT，CLEARTEXT，默认值为SSHA。

# Set password hashing algorithm to use by default

password-hash {SSHA}

# 动态载入模块，有两种方式：绝对路径或文件名，动态载入modulepath下的模块，该方式需在slapd编译时通过—enable-modules指明。

# Load dynamic backend modules:

# modulepath /usr/lib64/openldap

# Modules available in openldap-servers-overlays RPM package

# Module syncprov.la is now statically linked with slapd and there

# is no need to load it here

# moduleload accesslog.la

# moduleload auditlog.la

# moduleload denyop.la

# moduleload dyngroup.la

# moduleload dynlist.la

# moduleload lastmod.la

# moduleload pcache.la

# moduleload ppolicy.la

# moduleload refint.la

# moduleload retcode.la

# moduleload rwm.la

# moduleload smbk5pwd.la

# moduleload translucent.la

# moduleload unique.la

# moduleload valsort.la

# modules available in openldap-servers-sql RPM package:

# moduleload back\_sql.la

# TLS（连接加密）默认LDAP网络连接不加密。以文本方式传输，TLS通过X.509提供安全

# The next three lines allow use of TLS for encrypting connections using a

# dummy test certificate which you can generate by changing to

# /etc/pki/tls/certs, running "make slapd.pem", and fixing permissions on

# slapd.pem so that the ldap user or group can read it. Your client software

# may balk at self-signed certificates, however.

# TLSCACertificateFile /etc/pki/tls/certs/ca-bundle.crt

# TLSCertificateFile /etc/pki/tls/certs/slapd.pem

# TLSCertificateKeyFile /etc/pki/tls/certs/slapd.pem

# 指明通用ACLs

# Sample access control policy:

# Root DSE: allow anyone to read it

# Subschema (sub)entry DSE: allow anyone to read it

# Other DSEs:

# Allow self write access

# Allow authenticated users read access

# Allow anonymous users to authenticate

# Directives needed to implement policy:

# access to dn.base="" by \* read

# access to dn.base="cn=Subschema" by \* read

# access to \*

# by self write

# by users read

# by anonymous auth

#

# if no access controls are present, the default policy

# allows anyone and everyone to read anything but restricts

# updates to rootdn. (e.g., "access to \* by \* read")

#

# rootdn can always read and write EVERYTHING!

# 允许匿名登录。所有验证用户可以读取目录中任何数据。

# Global Read ACL

access to \*

by self write

by \* read

# 运行用户用ldappasswd更新自己的密码

# Change User Password ACL

access to attrs=userPassword

by self write

by anonymous auth

by \* none

# suffix是基础DN（Distinguished Name）

suffix "dc=it,dc=com"

# rootdn是完整DN

rootdn "cn=Manager,dc=it,dc=com"

# rootpw是rootdn的密码，用slappasswd生成加密密码写入slapd.conf

[root@node0 ~]# /usr/sbin/slappasswd

New password:

Re-enter new password:

{SSHA}qjdCJno9J4y8Sg0yNI57TYqWO4P0q8Jq

rootpw {SSHA}qjdCJno9J4y8Sg0yNI57TYqWO4P0q8Jq

# LDAP Directory Database，LDAP 目录树保存在目录数据库中，有两种类型数据库：Berkley DB（BDB）和HDB。HDB在大规模用户时效率高些。Database指明使用的目录数据库类型。Directory指明数据库备份地址。使用BDB需要DB\_CONFIG配置文件，将/etc/openldap/DB\_CONFIG.example拷贝到LDAP目录数据库中。

database bdb

# The database directory MUST exist prior to running slapd AND

# should only be accessible by the slapd and slap tools.

# Mode 700 recommended.

directory /var/lib/ldap //index存储位置，需设置正确权限供slapd读写

## 命令

在执行slapadd，slapcat或slapindex前必须先停止slapd服务，否则将破坏数据库完整性

### slapd

-f filename配置文件路径，默认为/etc/openldap/slapd.conf；

-F slapd-config-directory配置目录路径，默认为/usr/local/etc/openldap/slapd.d；

如果-f和-F同时使用，配置文件将转换为配置目录形式。

-h URLs监听配置，默认为ldap:///，389端口，可指定其他协议，如-h ldaps:// -h ldap://127.0.0.1:667，生成2个监听：监听LDAP/SSL的636端口，监听LDAP/TCP的localhost的667端口（须在/etc/sysconfig/ldap设置）；

-n server-name，默认为slapd；

-l syslog-local-user为syslog指定user，默认为LOCAL4；

-u user –g group 指定user和group，可为名称或id

-r directory 指定运行时目录

-d level 调试等级

| Level | Description |
| --- | --- |
| -1 | enable all debugging |
| 0 | no debugging |
| 1 | trace function calls |
| 2 | debug packet handling |
| 4 | heavy trace debugging |
| 8 | connection management |
| 16 | print out packets sent and received |
| 32 | search filter processing |
| 64 | configuration file processing |
| 128 | access control list processing |
| 256 | stats log connections/operations/results |
| 512 | stats log entries sent |
| 1024 | print communication with shell backends |
| 2048 | print entry parsing debugging |

### slapadd

slapadd -l <inputfile> -f <slapdconfigfile> [-d <debuglevel>] [-n <integer>|-b <suffix>]

[root@node0 ~]# slapadd -l base.ldif -n 2 //向第二个数据库中添加base.ldif

-l inputfile：LDIF输入文件位置

-f slapdconfigfile：slapd.conf文件位置，索引位置和使用哪些索引

-d debuglevel：调试等级

-n databasenumber：修改哪个数据库，配置文件中的第一个为1，第二个为2，依次推类。默认为1，不能与-b同用

-b suffix：修改哪个数据库，提供的后缀须与slapd.conf中的suffix符合。不能与-n同用。

### slapcat

[root@node0 ~]# slapcat -l base.ldif -n 2 //将第二个数据库中的内容导出到base.ldif中

### slapindex

[root@node0 ~]# service slapd stop

[root@node0 ~]# slapindex -n 2 //当修改/etc/openldap/slapd.conf后，重新索引slapd目录

### slappasswd

[root@node0 ~]# slappasswd //为rootpw或ldapmodify生成加密密码

**以下为openldap-clients的程序：**

### ldapadd

增加LDAP记录，是ldapmodify –a的硬链接

-w 密码

-W 提示输入密码Enter LDAP Password:

-f 文件路径

### ldapdelete

删除LDAP记录

ldapdelete [-n] [-v] [-k] [-K]

[-c] [-d debuglevel] [-f file] [-D binddn]

[-W] [-w passwd] [-h ldaphost] [-p ldapport]

[dn]...

### ldapmodify

修改记录

ldapmodify [-a] [-b] [-c] [-r]

[-n] [-v] [-k] [-d debuglevel]

[-D binddn] [-W] [-w passwd]

[-h ldaphost] [-p ldapport] [-f file]

ldapadd [-b] [-c] [-r] [-n]

[-v] [-k] [-K] [-d debuglevel]

[-D binddn] [-w passwd] [-h ldaphost]

[-p ldapport] [-f file]

### ldappasswd

LDAP用户设置密码

### ldapsearch

搜索数据库记录，filter用于过滤记录内容

ldapsearch [-n] [-u] [-v] [-k]

[-K] [-t] [-A] [-B] [-L]

[-R] [-d debuglevel] [-F sep] [-f file]

[-x] [-D binddn] [-W] [-w bindpasswd]

[-h ldaphost] [-p ldapport] [-b searchbase]

[-s base|one|sub]

[-a never|always|search|find] [-l timelimit]

[-z sizelimit] filter [attrs...]

[root@node0 ~]# ldapsearch -x -u -b 'dc=scs,dc=com'

-u以友好方式输出，-x使用simple authentication，-b是搜索路径

### ldapcompare

连接到LDAP server，对比

### ldapmodrdn

连接到LDAP server，修改RDN

## 端口类型

Slapd监听ldap://的389/tcp端口，和ldaps://的636/tcp端口。

LDAP三种认证方式：

* anonymous：取消匿名登录，须在slapd.conf中指明disallow bind\_anon，但不能阻止匿名访问目录，需要认证访问目录，须指明require authc。
* unauthenticated：未经认证绑定默认取消，可在slapd.conf中指明allow bind\_anon\_cred启用。
* user/password authenticated：默认启用，然而密码是明文传送，建立用TLS保护，须设置disallow bind\_simple，从而取消简单认证。

|  |  |  |
| --- | --- | --- |
| **URL** | **Protocol** | **Transport** |
| ldap:/// | LDAP 不加密 | TCP port 389 |
| ldaps:/// | LDAP over SSL | TCP port 636 |
| ldapi:/// | LDAP | IPC (Unix-domain socket) |

## LDIF

#comment

dn: <distinguished name>

<attrdesc>: <attrvalue>

<attrdesc>: <attrvalue>

...

attributes：values用于描述记录，objectclass定义记录所属的objectclass，所有记录必须包含一个objectclass属性，表明其属于哪个集合， LDAP提供以下基本objectclass：

* Groups
* Locations，例如国家名和描述
* Organizations：企业
* People：个人，包括电话，邮件等。

特殊字符用双冒号转成Base64

<attrdesc>::<attrvalue>

## OpenSSL/TLS

SSL(Secure Socket Layer) 安全协议，由 Netscape 公司首先提出，最初用在保护 Navigator 浏览器和 Web 服务器之间的 HTTP 通信 ( 即 HTTPS)，SSL被 IETF 吸收改进为 TLS(Transport Layer Security) 协议,也称为SSL/TLS协议。

SSL生成证书流程

|  |
| --- |
| server生成私钥server.key--> 生成csr -> +（ca.key+ca.crt）-> 生成证书server.crt                                       / CA导出私钥(ca.key)和证书(ca.crt)                                       \ client生成私钥client.cky--> 生成csr -> +（ca.key+ca.crt）-> 生成证书client.crt |

[root@node0 misc]# openssl genrsa -des3 -out server.key 1024 //生成服务器私钥

[root@node0 misc]# openssl rsa -in server.key -out server.key //去除私钥密码

[root@node0 misc]# openssl req -new -key server.key -out server.csr  //利用私钥生成证书

[root@node0 misc]# openssl genrsa -des3 -out client.key 1024 //生成客户端私钥

[root@node0 misc]# openssl req -new -key client.key -out client.csr //生成客户端证书

[root@node0 misc]# openssl rsa -in client.key -out client.key

[root@node0 misc]# openssl req -new -x509 -keyout ca.key -out ca.crt //制作CA证书和私钥

[root@node0 misc]# openssl rsa -in ca.key -out ca.key

//用生成的CA的证书和CA私钥(ca.crt,ca.key)为刚才生成的server.csr,client.csr文件签名，从而作成服务器证书server.crt和客户端证书client.crt:

[root@node0 misc]# mkdir -p CA/crl

[root@node0 misc]# mkdir -p CA/newcerts

[root@node0 misc]# mkdir -p CA/private

[root@node0 misc]# touch CA/index.txt

[root@node0 misc]# echo 01 > CA/serial

[root@node0 misc]# rm -f /etc/pki/CA

[root@node0 misc]# mv CA/ /etc/pki/CA

[root@node0 misc]# openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key

Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y

[root@node0 ~]# cd /etc/pki/tls/misc

[root@node0 misc]# ./CA -newca **//创建主证书,PEM（Privacy Enhanced Mail）是加密格式**

//如果执行出错，则rm -f /etc/pki/CA/private/cakey.pem

CA certificate filename (or enter to create) //回车

Enter PEM pass phrase: //输入密码

Verifying - Enter PEM pass phrase: //重复密码

Country Name (2 letter code) [AU]:cn

State or Province Name (full name) [Some-State]:cn

Locality Name (eg, city) []:cn

Organization Name (eg, company) [Internet Widgits Pty Ltd]:cn

Organizational Unit Name (eg, section) []:cn

Common Name (eg, YOUR name) []:localhost

Email Address []:05251152@bjtu.edu.cn

Enter pass phrase for /etc/pki/CA/private/./cakey.pem: //再次输入以上密码

[root@node0 certs]# cp server.csr /etc/pki/tls/misc/newreq.pem //将证书放到CA同目录中，重命名为newreq.pem

[root@node0 certs]# cd /etc/pki/tls/misc/

[root@node0 misc]# ./CA -sign //为服务器证书签名，生成newcert.pem

[root@node0 misc]# cp newcert.pem /etc/pki/tls/certs/server.csr //覆盖原来的服务器证书

/usr/libexec/openldap/generate-server-cert.sh

<http://blog.sina.com.cn/s/blog_6151984a0100f0q4.html>

**问题1：failed to update database TXT\_DB error number 2**

[root@node0 CA]# rm -f /etc/pki/CA/index.txt

[root@node0 CA]# touch /etc/pki/CA/index.txt

## 参考网页

<http://www.storageonline.com.cn/rec-posts/the-ldap-authentication-the-red-hat-an-an-enterprise-linux-centos-6-2/>

<http://www.openldap.org/doc/admin24/quickstart.html>

<http://jamyy.dyndns.org/blog/2012/01/3506.html>

<http://www.tldp.org/HOWTO/LDAP-HOWTO/index.html>

# OpenLDAP for JAVA

## 使用手册

String MY\_HOST = "localhost";

int MY\_PORT = 389;

int MY\_SOCKET\_TIMEOUT = 20000;

LDAPConnection ld = new LDAPConnection (MY\_SOCKET\_TIMEOUT); //设置连接超时

ld.connect(MY\_HOST, MY\_PORT); //连到HOST和PORT上

ld.bind(loginDN, loginPW); //如果loginDN 为NULL或””，则匿名连接

### 限制操作

两种类型

* **Connection-based**：限制应用于所有针对该连接的操作；

LDAPConnection lc = new LDAPConnection();

LDAPConstraints cons = lc.getConstraints(); //获得先前设置的限制

cons.setTimeLimit(5000);

lc.setConstraints(cons);

* **Operation-based**：只针对当前操作有效。

LDAPConnection lc = new LDAPConnection();

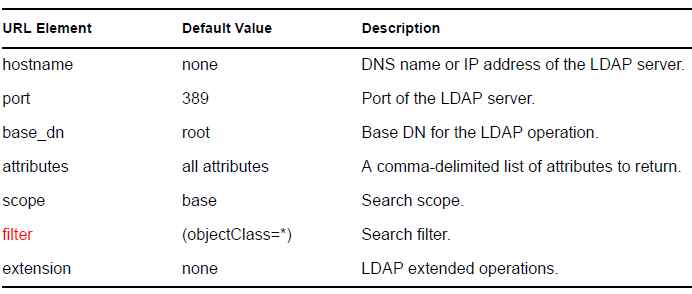
LDAPConstraints cons = lc.getConstraints();

cons.setTimeLimit(5000);

lc.add(entry, cons); //在操作中设置限制，即在增加该entry时，设置最大时间限制

### LDAP URLs

ldap[s]://<hostname>:<port>/<base\_dn>?<attributes>?<scope>?<filter>?<extension>



### LDAPConnection.search

5个参数：

* Search Base：搜索区域dn，为””则从根节点开始搜索。
* Search Scope：搜索深度，有四个选项：SCOPE\_BASE，只搜索在Search Base中的记录，在知道记录的Search Base，而且需要获得其属性时使用；SCOPE\_ONE，搜索Search Base下一层记录，例如o=acme作为Search Base时，返回其包含的记录，但不包括o=acem；SCOPE\_SUB，所有Search Base下的记录，包括Search Base；SCOPE\_SUBORDINATESUBTREE，与SCOPE\_SUB相同。
* Search Filter：过滤记录。例如title=engineer搜索所有title为enginner的记录。
* Attribute List：返回记录的属性列表，默认值null返回全部属性。String[] attrList = new String[]{"\*", "creatorsName"}。\*或ALL\_USER\_ATTRIBUTES。
* Types Only：是否返回属性列表和属性值，FLASE返回属性和值。

#### synchronous search

String searchBase = "ou=development,o=acme";

int searchScope = LDAPConnection.SCOPE\_BASE;

String searchFilter = "(title=engineer)";

//返回所有title=engineer在development acme中的记录

LDAPSearchResults searchResults = lc.search( searchBase,searchScope,searchFilter,null,false);

//对LDAPSearchResults遍历

while (searchResults.hasMore()) {

LDAPEntry nextEntry = null

nextEntry = searchResults.next();

System.out.println("\n We found "+nextEntry.getDN());

LDAPAttributeSet attributeSet = nextEntry.getAttributeSet();

Enumeration allAttributes = attributeSet.getAttributes();

while(allAttributes.hasMore()) {

LDAPAttribute attribute =

(LDAPAttribute)allAttributes.nextElement();

String attributeName = attribute.getName();

System.out.println("\t\t"+ attributeName);

Enumeration allValues = attribute.getStringValues();

if(allValues!= null) {

while(allValues.hasMore()) {

String Value = (String) allValues.nextElement();

System.out.println("\t\t\t" + Value);

}

}

}

#### asynchronous search

String searchBase = "ou=development,o=acme";

int searchScope = LDAPConnection.SCOPE\_BASE;

String searchFilter = "(title=engineer)";

LDAPSearchListener listener =

lc.search(searchBase,searchScope,searchFilter,null,false(LDAPSearchListener)null,(LDAPSearchConstraints)null);

//LDAPMessage有三种类型：LDAPSearchResult，一条记录；LDAPSearchResultReference，连续记录的引用；LDAPResponse：搜索结束标志，可为SUCCESS，REFERRAL。

LDAPMessage message;

while ((message = listener.getResponse())!= null) {

//判断是哪种类型

if (message instanceof LDAPSearchResultReference) {

}

}

### Exception Handling

两种：LDAPException，LDAPReferralException

1. Synchronous Methods

同步方法中，用try/catch处理LDAPException

try {

[Attempt an LDAP operation]

}

catch(LDAPException e) {

System.out.println("Error:" + e.toString());

}

//通过设置BatchSize为0（默认为1），收到所有记录并保持到本地内存才返回

LDAPConnection ld = new LDAPConnection();

LDAPSearchConstraints cons = ld.getSearchConstraints();

cons.setBatchSize(0);

ld.setConstraints(cons);

1. Asynchronous Methods

异步方法中，只会抛出本地/非服务器错误，如连接或参数错误。服务器错误通过LDAPResponse.getResultCode获得。

//previously determined that the message is an LDAPResponse

if (message instanceof LDAPResponse)

{

LDAPResponse response = (LDAPResponse) message;

int status = response.getResultCode();

if (status!= LDAPException.SUCCESS)

{

System.out.println("Asynchronous search failed.");

throw new LDAPException(response.getErrorMessage(),status,response.getMatchedDN());

}

}

1. Referral Exceptions

LDAPReferralExceptions当引用意外关闭或沿着引用查询结果时抛出，继承自LDAPException，包含一系列URL。

try {

[perform an LDAP search operation here]

}

catch(LDAPReferralException e) {

LDAPUrl urls[] = e.getURLs();

System.out.println("Referral Exception");

for(i=0;i < urls.length;i++)

System.out.println(urls[i];

}

### 增加Entry

//生成实体属性，加入属性集合

LDAPAttribute attribute = null;

String cn\_values[] = {"James Smith", "Jim Smith", "Jimmy Smith"};

attribute = new LDAPAttribute("cn", cn\_values);

attributeSet.add(attribute);

//生成LDAPEntry，设置实体DN，和属性

String dn = "cn=JSmith," + containerName;

LDAPEntry newEntry = new LDAPEntry(dn, attributeSet);

//调用LDAPConnection，将LDAPEntry增加到目录中

lc.connect(ldapHost, ldapPort);

lc.bind(ldapVersion, loginDN, password);

lc.add(newEntry);

### 读Root DSE

int ldapVersion = LDAPConnection.LDAP\_V3;

LDAPConnection lc = new LDAPConnection();

lc.connect(LDAP Host, ldapPort);

lc.bind(ldapVersion, loginDN, password);

//设置Search Base为””，Search Filter为objectclass=\*，Search Scope为LDAP\_SCOPE\_BASE

LDAPSearchResults searchResults = lc.search( "",LDAPConnection.SCOPE\_BASE,"(objectclass=\*)",

returnedAttributes,attributeOnly);

### 读Schema

//连接LDAP server

LDAPConnection lc = new LDAPConnection();

lc.connect([LDAP Host], [ldapPort]);

lc.bind(ldapVersion, [loginDN], [password]);

//获得schema

LDAPSchema schema = lc.fetchSchema(getSchema = lc.getSchemaDN());

//读取属性和类

Enumeration enum = schema.getAttributes()

while(enum.hasMoreElements())

{

LDAPAttributeSchema attr =

(LDAPAttributeSchema)enum.nextElement();

System.out.println(attr.getNames()[0]);

System.out.println(attr.getSyntaxString());

}

enum = schema.getObjectClasses();

while(enum.hasMoreElements())

{

LDAPObjectClassSchema objcls

(LDAPObjectClassSchema)enum.nextElement();

System.out.println(objcls.getNames()[0]);

}

## 项目开发

新建工程，将novell-jldap-devel-2009.10.07-1unix目录中的lib和samples目录拷贝到工程中，Project->Properties->libraries中增加ldap.jar和utilities.jar，下载axis的jaxrpc.jar和tomcat的servlet-api.jar加入工程

Run as->Run Configuration->Arguments设置Program Arguments

account 需要uid，inetOrgPerson需要sn

AddEntry：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456 "ou=People,dc=scs,dc=com"

DeleteEntry：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456 "cn=JSmith,ou=People,dc=scs,dc=com"

GraceLogin：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456

List：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456 "dc=scs,dc=com" "(objectclass=\*)"

ListGroup：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456 "ou=Company,dc=scs,dc=com"

ListSchema：ApplicationArguments和Argument用于解析参数

Search，ClientSideSort：192.168.0.59 "cn=Manager,dc=scs,dc=com" 123456 "ou=Group,dc=scs,dc=com" "(objectclass=\*)"

**问题1：Access restriction: The type Provider is not accessible due to restriction on required library**

Project->Properties->libraries，先remove掉JRE System Library，在Add Library重新加入。

**问题2：Server Message: no structural object class provided**

Objectclass为记录所属的若干个类型，如posixAccount，posixAccount中定义了一系列属性（必要和非必要），新增记录时需设置好必要属性，posixAccount必须与其他objectclass搭配使用，如account，inetOrgPerson或organizationalPerson，否则会报该错误。