一 、什么是LDAP

LDAP：(轻量级目录访问协议，Lightweight Directory AccessProtocol）

       它是基于 X.500标准的，但是简单多了并且可以根据需要定制。与X.500不同，LDAP支持TCP/IP，这对访问Internet是必须的。

目录是一个为查询、浏览和搜索而优化的专业分布式数据库，它成树状结构组织数据，就好 象Linux/Unix系统中的文件目录一样。目录数据库和关系数据 库不同，它有优异的读性能，但写性能差，并且没有事务处理、回滚等复杂功能，数据修改使用简单的锁定机制实现All-or-Nothing,不适于存储修 改频繁的数据。所以目录天生是用来查询的，就好象它的名字一样。现在国际上的目录服务标准有两个，一个是较早的X.500标准，一个是较新的LDAP标 准。  
       LDAP诞生的目标是快速响应和大容量查询并且提供多目录服务器的信息复制功能，它为读密集型的操作进行专门的 优化。因此，当从LDAP服务器中读取数据的时候会比从专门为OLTP优化的关系型数据库中读取数据快一个数量级。

LDAP常用术语解释：  
DN：distinguished name。在LDAP目录中的所有记录项都有一个唯一的DN  
CN,OU,DC都是LDAP连接服务器的端字 符串中的区别名称;  
LDAP连接服务器的连接字串格式为：ldap://servername/DN    
其中DN有三个属性，分别是CN,OU,DC    
LDAP是一种通讯协议，如同HTTP是一种协议一样的！在LDAP目录中。  
uid    (User ID) CN     (Common   Name)  DC     (Domain   Component)   域名元素       OU     (Organizational   Unit)  SN    （surname)   
An LDAP 目录类似于文件系统目录.     下列目录: DC=redmond,DC=wa,DC=microsoft,DC=com  如果我们类比文件系统的话，可被看作如下文件路径:     
Com/Microsoft/Wa/Redmond

例如：CN=test,OU=developer,DC=domainname,DC=com  
在上面的代码中cn=test代表一个用户名，ou=developer代表一个active directory中的组织单位。  
这句话的含义是test这个对象处在domainname.com域的developer组织单元中。

二、LDAP的安装

1.下载安装 openldap for windows，版本2.2.29.可以在百度上搜索openldap for windows。

        安装很简单，一路 next 即可，假设我们安装在C:\Program Files\OpenLDAP  
2.配置 openldap，编辑 slapd.conf 文件

1)、打开C:\Program Files\OpenLDAP \slapd.conf，找到ucdata-path    ./ucdata  
include  ./schema/core.schema，在它后面添加  
include  ./schema/cosine.schema  
include  ./schema/inetorgperson.schema

接下来的例子只需要用到以上三个 schema，当然，如果你觉得需要的话，你可以把其他的 schema 全部添加进来

include         ./schema/nis.schema

include         ./schema/openldap.schema

include         ./schema/dyngroup.schema

include         ./schema/collective.schema

include         ./schema/corba.schema

include         ./schema/duaconf.schema

include         ./schema/java.schema

include         ./schema/misc.schema

include         ./schema/pmi.schema

include         ./schema/ppolicy.schema

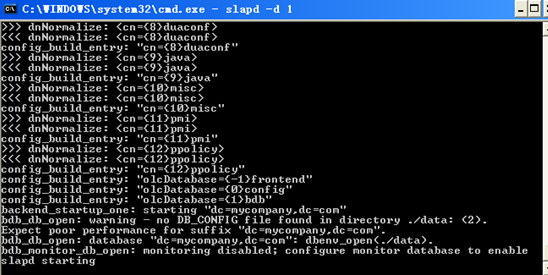
2)、下面我们做一个示例：  
     需要在 slapd.conf 配置文件中，找到  
     suffix  “dc=my-domain,dc=com”  
     rootdn  “cn=Manager,dc=my-domain,dc=com”  
    把这两行改为  
    suffix     "dc=mycompany,dc=com"

    rootdn   "cn=Manager,dc=mycompany,dc=com"  
  
suffix 就是看自己如何定义了，后面步骤的 ldif 文件就必须与它定义了。还要注意到这个配置文件中有一个 rootpw  secret，这个 secret 是cn=Manager 的密码，以后会用到，不过这里是明文密码，你可以用命令：slappasswd -h {MD5} -s secret  (“secret“是自定义的密码，可以随便设置)算出加密的密码{MD5}Xr4ilOzQ4PCOq3aQ0qbuaQ==替换配置中的 secret。

3. 启动 OpenLDAP  
    CMD 进入到C:\Program Files\OpenLDAP 下，  
1)、启动OpenLDAP-slapd服务：作用在于开机自动启动该服务项  
    slapd install OpenLDAP-slapd “OpenLDAP Directory Service” auto  
net start OpenLDAP-slapd  
NOTE: the “slapd install” is only needed if you didn’t choose the “create NTservice” option during installation.

当你完成这一步后，下次开机的时候就自动启动LDAP的服务了，不必再次手动启动。（此步也可以忽略不做！）  
2)、启 动OpenLDAP服务器  
在cmd下运行：slapd -d 1

启动成功可以看到控制台下打印一片信息，如下图



openldap 默认是用的 Berkeley DB 数据库存储目录数据的。

4. 建立条目,编辑导入 ldif 文件1) 新建一个 ldif(LDAP Data Interchanged Format) 文件(纯文本格式)，例如 setup.ldif.

setup.ldif 文件内容如下：

dn: dc=train,dc=com

objectClass: top

objectClass: dcObject

objectClass: domain

dc: train

dn: ou=roles,dc=train,dc=com

objectClass: top

objectClass: organizationalUnit

ou: roles

dn: ou=people,dc=train,dc=com

objectClass: top

objectClass: organizationalUnit

ou: people

dn: cn=Test Users,ou=roles,dc=train,dc=com

objectClass: groupOfUniqueNames

cn: Test Users

uniqueMember: uid=sspecial,ou=people,dc=train,dc=com

uniqueMember: uid=jbloggs,ou=people,dc=train,dc=com

uniqueMember: uid=hzhangse,ou=people,dc=train,dc=com

dn: cn=Special Users,ou=roles,dc=train,dc=com

objectClass: groupOfUniqueNames

cn: Special Users

uniqueMember: uid=sspecial,ou=people,dc=train,dc=com

uniqueMember: uid=hzhangse,ou=people,dc=train,dc=com

dn: cn=Admin Users,ou=roles,dc=train,dc=com

objectClass: groupOfUniqueNames

cn: Admin Users

uniqueMember: uid=admin,ou=people,dc=train,dc=com

uniqueMember: uid=hzhangse,ou=people,dc=train,dc=com

dn: uid=admin,ou=people,dc=train,dc=com

objectClass: person

objectClass: inetOrgPerson

cn: State App

displayName: App Admin

givenName: App

mail: admin@fake.org

sn: Admin

uid: admin

userPassword: adminpassword

dn: uid=jbloggs,ou=people,dc=train,dc=com

objectClass: person

objectClass: inetOrgPerson

cn: Joe Bloggs

displayName: Joe Bloggs

givenName: Joe

mail: jbloggs@fake.org

sn: Bloggs

uid: jbloggs

userPassword: password

dn: uid=sspecial,ou=people,dc=train,dc=com

objectClass: person

objectClass: inetOrgPerson

cn: Super Special

displayName: Super Special

givenName: Super

mail: sspecial@fake.org

sn: Special

uid: sspecial

userPassword: password

dn: uid=hzhangse,ou=people,dc=train,dc=com

objectClass: person

objectClass: inetOrgPerson

cn: Zhang Hong

displayName: Zhang Hong

givenName: Ryan

mail: 2387362072@qq.com

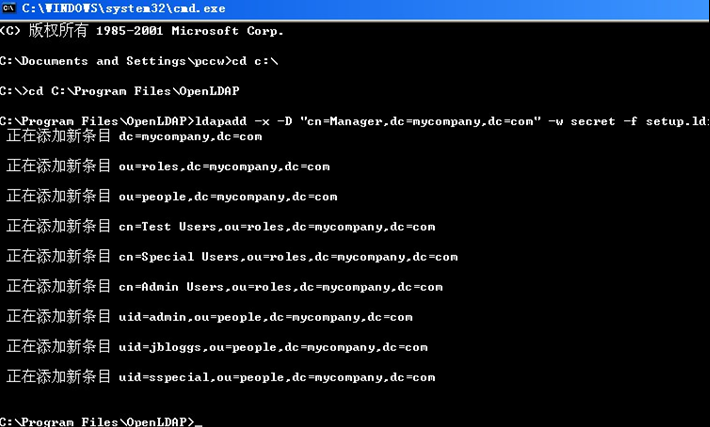
sn: zhanghong

uid: hzhangse

userPassword: hzhangse

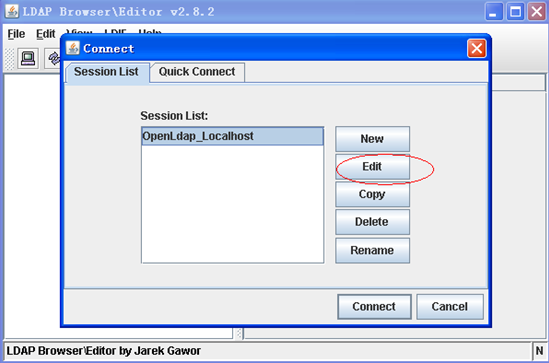
2) 执行命令：ldapadd -x -D“cn=manager,dc=mycompany,dc=com” -w secret -f setup.ldif

注意： ldapadd 命令使用的前提是必须装过oracle数据库。我之前没装，会提示ldapadd是不可执行的命令，总之就是系统不认识这个命令。后来装了oracle之后就可以用了。

****

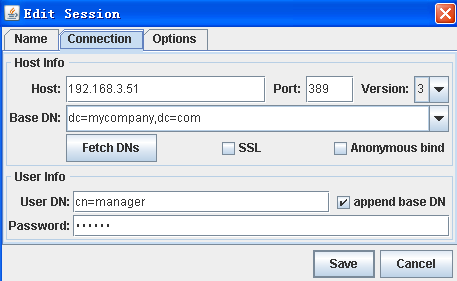
以上图示表示你已经将上述内容导入到LDAP数据库了。

5. LdapBrowser 浏览  
可点击链接 <http://www.blogjava.net/Files/Unmi/LdapBrowser282.rar>下载。解压并双击lbe.bat批处理文件，弹出LDAP Browser窗口中默认会有一个OpenLdap\_Localhost的链接，选edit

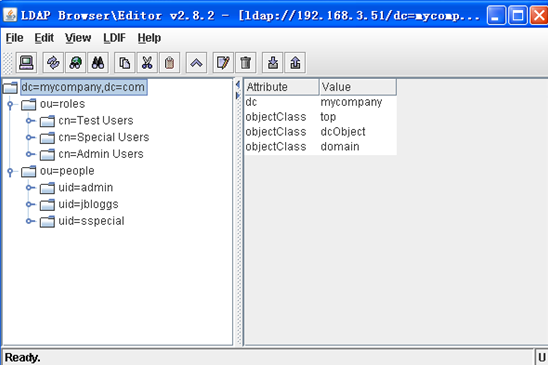
****

弹出下面的对话框，输入你的主机名：注意：如果是localhost 就不行。原因在哪里我也不很矛盾。输入你在slapd.conf文件中的DN，和密码。

SAVE—>>>Connect—>>>



连接成功之后看到的结构图：



6. Java 连接 openldap进行测试是否连接成功

**<span style="font-family:Arial;">package com.pccw.cms.test;**

**import java.util.Hashtable;**

**import javax.naming.Context;**

**import javax.naming.NamingException;**

**import javax.naming.directory.DirContext;**

**import javax.naming.directory.InitialDirContext;**

**public class LDAPTest {**

**public LDAPTest() {**

**}**

**@SuppressWarnings("unchecked")**

**public static void main(String[] args) {**

**String root = "dc=mycompany,dc=com"; //root**

**@SuppressWarnings("rawtypes")**

**Hashtable env = new Hashtable();**

**env.put(Context.INITIAL\_CONTEXT\_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");**

**env.put(Context.PROVIDER\_URL, "ldap://192.168.3.51/" + root);**

**env.put(Context.SECURITY\_AUTHENTICATION, "simple");**

**env.put(Context.SECURITY\_PRINCIPAL, "cn=manager,dc=mycompany,dc=com");**

**env.put(Context.SECURITY\_CREDENTIALS, "secret");**

**DirContext ctx = null;**

**try {**

**ctx = new InitialDirContext(env);**

**System.out.println("认证成功");**

**}**

**catch (javax.naming.AuthenticationException e) {**

**e.printStackTrace();**

**System.out.println("认证失败");**

**}**

**catch (Exception e) {**

**System.out.println("认证出错：");**

**e.printStackTrace();**

**}**

**if (ctx != null) {**

**try {**

**ctx.close();**

**}**

**catch (NamingException e) {**

**//ignore**

**}**

**}**

**System.exit(0);**

**}**

**} </span>**

7.启动OpenLDAP服务的批处理

新建一个文本文档，在文档写入一下内容：

@echo off

color 07

echo 正在启动HZCA-LDAP文件，请稍等........

cd  /d C:\Program Files\OpenLDAP

slapd -d 1

color 07

echo 启动失败！请重新启动OpenLDAP-slapd程序！输入启动指令：

slapd -d 1

echo.& pause

将文档保存为autoStart.bat 那么当你点击它是，就启动LDAP的服务了！（若你已经在前面将LDAP的服务设定为开机自动启动，那就不需要再次启动这个服务。）

好了，关于LDAP的基本配置就讲到这里。接下来的第二篇文章会讲述如何将LDAP与Tomcat整合配置，进行权限访问。

<RealmclassName="org.apache.catalina.realm.JNDIRealm"

   debug="99"

   connectionName="cn=Manager,dc=mycompany,dc=com"

   connectionPassword="secret"

   connectionURL="ldap://localhost:389"

   roleBase="ou=roles,dc=mycompany,dc=com"

   roleName="cn"

   roleSearch="(uniqueMember={0})"

   roleSubtree="false"

   userSearch="(uid={0})"

    userPassword="userPassword"

   userPattern="uid={0},ou=people,dc=mycompany,dc=com"

/>

使用方法：在D盘建立一个文件夹用于存放相应的文件比如：D:\LDAPTest

          将Tomcat文件包放入该文件目录D:\LDAPTest\apache-tomcat-6.0.30

          更改D:\LDAPTest\apache-tomcat-6.0.30\conf目录下server.xml

          将以上代码加入到 <Enginename="Catalina" defaultHost="localhost">  下面。

**2.Web 应用配置**

要完成 Tomcat 对 OpenLDAP 的配置, 应用的 *web.xml*文件必须更新的. 提供下载的应用中有 6 个 JSP 页面组成, 其中三个是针对LDAP 目录中设置的不同角色的受保护资源. 应用必须配置为允许基于 form 的验证，并被告知存在着哪些角色. 首先, 创建一个*login.jsp* 文件. 当使用基于 form 的验证时, 这个 JSP 必须包含如下内容.

<body>

    <formmethod="POST" action="j\_security\_check">

        <inputtype="text" name="j\_username">

        <br>

        <inputtype="password" name="j\_password">

        <br>

        <inputtype="submit">

    </form>

</body>

必须参考新的 OpenLDAP realm 来更新*web.xml* 文件, 并在其中使用 LDAP 指定的角色. 应用还需要让对*login.jsp* 页面的访问权限是公开的; 不然的话, 没有用户能登陆.

要告知应用什么角色能访问什么资源, 就要在应用 (或安全约束) 中用到 URL 映射. 这种映射既可以是一个文件名 (*/admin.jsp*) 也可以是一个路径 (/jsp/\* 将保护*jsp* 目录中的任何东西). 下面的 XML 表示列出的这两个.jsp 文件不受到保护.

<security-constraint>

   <web-resource-collection>

       <web-resource-name>Public Area</web-resource-name>

        <!--Define the context-relative URL(s) to be protected -->

       <url-pattern>/index.jsp</url-pattern>

       <url-pattern>/login.jsp</url-pattern>

    </web-resource-collection>

</security-constraint>

为什么这个代码会标记其他的 .jsp 文件为不受保护呢? 这是由于未列出的不受保护, 而非列出来的才不受保护. 下面的 *web.xml* 片断标明了 user.jsp 资源是受保护的, 并指定了什么角色可访问它.

<security-constraint>

   <web-resource-collection>

       <web-resource-name>Protected Area</web-resource-name>

        <!--Define the context-relative URL(s) to be protected -->

       <url-pattern>/user.jsp</url-pattern>

   </web-resource-collection>

   <auth-constraint>

    <!-- Anyonewith one of the listed roles may access this area -->

        <role-name>TestUsers</role-name>

       <role-name>Special Users</role-name>

       <role-name>Admin Users</role-name>

    </auth-constraint>

</security-constraint>

注意到那个受保护资源有还有另一个节点配置, <auth-constraint>, 它指定了什么应用角色可以访问上面的资源. 假如存在 <auth-constraint> 配置, 资源就会是安全的. 否则, 它们就是公开的. 除非整个应用都是受保护的 (例如, <url-pattern>/</url-pattern>), 公开的权限约束完全是多余.

欲配置 *web.xml* 文件利用前面创建的 *login.jsp* , 要添加下面的代码到 *web.xml* 文件中.

<!--  usesform-based authentication -->

<login-config>

   <auth-method>FORM</auth-method>

   <form-login-config>

       <form-login-page>/login.jsp</form-login-page>

       <form-error-page>/fail\_login.html</form-error-page>

   </form-login-config>

</login-config>

最后是对 *web.xml* 的一个附加配置步骤: 应用需要被知会我们要用到哪些角色.

<!-- Security roles referenced by this web application-->

<security-role>

   <role-name>Test Users</role-name>

</security-role>

<security-role>

   <role-name>Special Users</role-name>

</security-role>

<security-role>

   <role-name>Admin Users</role-name>

</security-role>

现在 Tomcat 已被配置为使用 OpenLDAP 并且我们的应用也对 *web.xml* 文件进行了正确的设置. 当用户首次被导航到 <security-constraint> 中列出的任何资源时 (只要不是公开的资源) , 服务器会自动显示*login.jsp* 页面要求用户验证. 假如用户验证失败,将会显示在 <form-error-page> 中指定的页面. 要是验证用户是通过了, 但是**未授权**访问某个资源 (比如, 不是某个角色的成员), 服务器会返回一个 403 错误页. 想要的话, 错误页可以在*web.xml* 文件中定制, 使用<error-code> 元素.

注,*web.xml* 中的元素有一个特定的顺序 (定义在 DTD 文件中), 所以你应该看看例子代码 中完整的*web.xml*文件. 尤其是 <login-config> 和 <security-role> 中的元素顺序.

现在用户可以登陆了, 但是该如何登出呢? 可以写代码来让用户的会话失效, 或者我们可以使用 Apache Jakarta 项目提供的便利的会话标签库.

**3.Jakarta 相应标签库**

通过加入 *logout.jsp* 到应用中并使用 Apache Jakarta Project 的会话标签库 , 我们能在不写任何定制代码情况下就失效掉的用户 (在实际的应用中, 你也许还需为清除一个用户会话做更多一些事情, 你也可能不怎么满意这么做).

<body>

   <sess:invalidate/>

    You are nowlogged out<br>

    <ahref="index.jsp">Return to index</a>

</body>

一旦包含用 <sess:invalidate/> 标签的 JSP 页面显示之后, 用户会话便被移除, 该用户被有效的注销掉了. 简单的放置一个链接指向*logout.jsp* 页面, 就能让用户导航该页, 这足以应付简单的应用. 另, 除会话标签库外, 我们还能利用 request 标签库来定制化基于用户和角色 JSP 页面的内容.

<req:isUserInRole role="Admin Users">

    The remote useris in role "Admin Users".<br />

</req:isUserInRole>

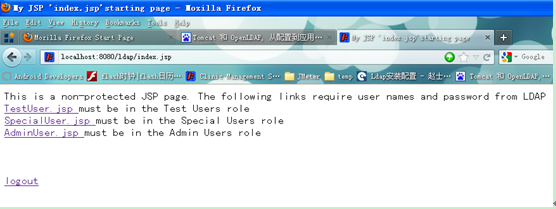
前述的 JSP 片断只有在当前用户在请求的角色中时才显示内容. 要验证角色安全性的行为, 可以用 JXplorer 快速的从角色中添加和移除用户.

**4．使用web应该测试权限**

 方法：将配置好的ldap应用放入D:\LDAPTest\apache-tomcat-6.0.30\webapps

    目录下。并启动Tomcat服务。

   在地址栏输入：http://localhost:8080/ldap/index.jsp



以上就是关于访问权限的配置，有一部分还是借鉴了别人的，如有不足之处，还请批评改正。谢谢~