据。在 LDAP 中共有 4 类 10 种操作。

① 查询类操作，如搜索、比较。

② 更新类操作，如添加条目、删除条目、修改条目、修改条目名。

③ 认证类操作，如绑定、解绑定。

④ 其他操作，如放弃和扩展操作。

除了扩展操作，另外 9 种是 LDAP 的标准操作。扩展操作是 LDAP 中为了增加新的功能提供的一种标准的扩展框架，当前已经成为 LDAP 标准的扩展操作，有修改密码和 StartTLS 扩展，在新的 RFC 标准和草案中正在增加一些新的扩展操作，不同的 LDAP 厂商也均定义了自己的扩展操作。

4．安全模型

安全模型提供一个安全框架，保护目录中的信息不被非法访问。LDAP 中的安全模型主要通过身份认证、安全通道和访问控制（ACL）实现。

LDAP 是一个面向连接的协议，在能够对 LDAP 目录进行任何操作之前，LDAP 客户端必须获得一个到 LDAP 服务端的连接，在这个过程中需要对 LDAP 客户端的身份进行验证，这一过程可以理解为用户绑定。

LDAP v2 只支持简单的密码验证。LDAP v3 实现了 SASL 安全框架，SASL 为多种验证协议提供了一种标准的验证方法，对于不同的验证系统，可以实现特定的 SASL 机制。

在 LDAP 中提供了基于 SSL/TLS 的通信安全保障。SSL/TLS 基于 PKI 信息安全技术，LDAP 通过 StartTLS 方式启动 TLS 服务，可以提供通信中的数据保密性、完整性保护；通过强制客户端证书认证的 TLS 服务，同时可以实现对客户端身份和服务器端身份的双向验证。

在用户通过验证之后，可以为该用户分配附加的权限。比如，一些用户只能查看特定的条目，而不能修改；一些用户可以查看并且修改所有的条目等。这一过程可以理解为访问控制。

6.1.5 LDAP Schema

目录 Schema 是一个规则集，定义了 LDAP 目录所应遵循的结构和规则，它决定哪些信息可以存放在目录服务中，以及在客户端和目录服务器查询交互中如何处理信息。目录服务器在存储新数据或修改现有数据时，会检查数据是否满足 schema 规则，当客户端或服务器比较两个属性值时，它们会使用 schema 规定的比较算法。

如果使用过关系型数据库，那么对模式应该不会陌生。关系数据库系统都是通过表格的形式进行数据存储的。在这之前，要首先定义表结构，即模式。表结构由一些字段组成，每个字段都有一个类型，以及一些约束条件。这就规定了我们可以存储的信息。

不过与关系型数据库系统不同的是，作为 LDAP 目录服务器的用户而言，一般不需要自己定义模式，所有实现 LDAP 协议的目录服务器都已经定义好了许多模式，这些模式可以解决大部分的信息存储问题。很多常用 schema 都在 RFC2252 中进行了定义，主流 LDAP 服务器都会支持这些基本的 schema。

在 LDAP 的 schema 中，有 4 个重要元素：对象类、属性、语法、匹配规则。

1．对象类（objectClass）

objectClass 定义了一个对象类，它说明了该对象类有哪些属性，哪些属性是必需的，哪些又是可选的。一个对象类的定义包括名称、描述、类型（包括结构类型或辅助类型）、必需属性、可选属性等信息。

2．属性（attribute）

属性就是一个对象类中可能包含的属性，对其的定义包括名称、数据类型、单值还是多值以及匹配规则等。

3．语法（Syntax）

语法是 LDAP 中会用到的数据类型和数据约束，它遵从 X.500 中数据约束的定义。其定义需要有一个 ID（遵从 X.500）以及说明（DESP）。

4．匹配规则（Matching Rules）

下面列举出了一些 LDAP 规定好的或是现在比较通用的 schema，一般的 LDAP 服务器都应该可以识别这些定义。

下面是一个名为 subschema 的对象类的定义：

```
(2.5.20.1 NAME 'subschema'  AUXILIARY
    MAY ( dITStructureRules $ nameForms $ ditContentRules $
    objectClasses $ attributeTypes $ matchingRules $ matchingRuleUse ))
```

首先是 OID，这里是 2.5.20.1，接着是 NAME 名为 subschema，AUXILIARY 说明是辅助型，之后是可选属性的定义，subschema 中没有定义必需属性，如果需要定义，应该和 MAY 一样，将属性放在 MUST（）中并用$隔开。

再来看一个属性 cn 的定义：

```
( 2.5.4.3 NAME 'cn' SUP name EQUALITY caseIgnoreMatch )
```

首先是 OID，这里是 2.5.4.3，接着是 NAME 名为 cn，再接着是父属性，名为 name，使用 caseIgnoreMatch 进行相等性（EQUALITY）匹配。

语法定义一般都比较简单，如：

```
( 1.3.6.1.4.1.1466.115.121.1.6  DESC  'String' )
```

这个定义说明，这一串数字 1.3.6.1.4.1.1466.115.121.1.5 代表了 LDAP 中的字符串，这个数字串的定义和 X.500 相关，包括了它的存储方式、所占空间大小等。

最后是匹配规则的例子，下面是 caseIgnoreMatch 的匹配规则：

```
( 2.5.13.2 NAME 'caseIgnoreMatch' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15)
```

首先是 OID，这里是 2.5.13.2，接着是 NAME 名 caseIgnoreMatch，再接着是语法，规定了数据类型，1.3.6.1.4.1.1466.115.121.1.15 就是 LDAP 数据类型 DirectoryString 的 OID，说明应用'caseIgnoreMatch'匹配规则的属性的类型必须是 DirectoryString 这个数据

类型才有效。

表 6-3 给出了 LDAP 协议中定义的一些常用属性及其含义(具体信息见 RFC 2252 文档)。

表 6-3 LDAP 协议定义的常用属性

属性	中文名称	描述
c	国家名称	值为两位国家代码，例如，中国：CN 美国：US
cn	通用名称	
dc	域名组件	如 sohu.com、dc=sohu, dc=com
co	国家名称	国家的全名
gn	givenName	
homephone	家庭电话号码	
mail	邮件地址	
mobile	移动电话号码	
o	组织名称	
ou	部门名称	通常为组织机构下的一个部门或者一个大型实体下的一个子实体
postalCode	邮政编码	
sn	姓，别名	
st	州或者省的名称	
street	街道地址	
userPassword	用户密码	
uid	用户 ID	
departmentNumber	部门编号	
displayName	显示名称	
description	描述	
employeeNumber	员工编号	
manager	经理	

表 6-4 是一些 LDAP 协议中定义的对象类（具体信息查阅 RFC 2252 文档）。

表 6-4 LDAP 协议定义的常用对象类

对 象 类	必需属性	可 选 属 性
account	userid	description $ seeAlso $ localityName $ organizationName $ organizationalUnitName $ host
country	c	searchGuide $ description
dcObject	dc	
device	cn	serialNumber $ seeAlso $ owner $ ou $ o $ l $ description
inetOrgPerson->person 继承 person		audio $ businessCategory $ carLicense $ departmentNumber $ displayName $ employeeNumber $ employeeType $ givenName $ homePhone $ homePostalAddress $ initials $ jpegPhoto $ labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $ roomNumber $ secretary $ uid $ userCertificate $ x500uniqueIdentifier $ preferredLanguage $ userSMIMECertificate $ userPKCS12

（续表）

对　象　类	必需属性	可　选　属　性
organizationalPerson 继承 Person		title $ x121Address $ registeredAddress $ destinationIndicator $ preferredDelivery Method $ telexNumber $ teletexTerminalIdentifier $ telephoneNumber $ internationali SDNNumber $ facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $ postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l
organization	o	userPassword $ searchGuide $ seeAlso $ businessCategory $ x121Address $ registeredAddress $ destinationIndicator $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $ facsimile TelephoneNumber $ street $ postOfficeBox $ postalCode $ postalAddress $ physical DeliveryOfficeName $ st $ l $ description
organizationalRole	cn	x121Address $ registeredAddress $ destinationIndicator $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $ facsimileTelephoneNumber $ seeAlso $ roleOccupant $ preferredDeliveryMethod $ street $ postOfficeBox $ postalCode $ postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l $ description
organizationalUnit	ou	userPassword $ searchGuide $ seeAlso $ businessCategory $ x121Address $ registeredAddress $ destinationIndicator $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $ facsimile TelephoneNumber $ street $ postOfficeBox $ postalCode $ postalAddress $ physical DeliveryOfficeName $ st $ l $ description
Person	cn　sn	userPassword $ telephoneNumber $ seeAlso $ description
Top（所有类的基类）		

6.1.6　LDAP 认证方式

LDAP 提供多种认证策略，包括匿名认证、简单口令、基于 SASL 的散列认证、基于 SSL 的简单口令、基于 SSL 的双向证书认证等。

① 匿名认证。在某些场景下，需把数据共享给目录下的所有用户，此时不需要对用户身份进行认证。

② 简单口令。由客户端向服务端发送身份和口令，完成身份验证，是最简单的认证方式，但存在安全风险。

③ 基于 SASL 的散列认证。与简单口令相比，客户端不向服务器发送口令，而是服务器向客户端发送挑战码，客户端以其知道的秘密信息对挑战码进行处理后返回给服务器端，服务器据此判断用户身份。

④ 基于 SSL 的简单口令。与简单口令相比，数据的传输在 SSL 通道上进行，防止了口令的泄密。

⑤ 基于 SSL 的双向证书认证。基于 SSL 的数字证书认证具有最高的安全级别，能够保证数据的保密性、数据的完整性，并且利用数字签名技术对客户端身份进行验证。

在选择完认证策略后，必须使用访问控制列表对用户可以访问的数据进行控制。每个访问控制列表指定三个方面的内容。

① 目录中的一个或多个资源：资源又称为对象，是访问控制的目标，通常包括条目、

属性、属性值。

② 一个或多个主体：主体是拒绝或同意访问的条目，主体可以用目录条目的名称或描述性信息指定。

③ 一个或多个访问权限：访问权限决定主体能够操作的资源。例如，访问权限是“搜索和读取”，表明主体可查询并读取条目的属性。

由于不同目录服务的实现有差异，我们以 OpenLDAP 的访问控制为例。首先看一下 ACL 访问指令的格式：

```
<access directive> ::= access to <what> [by <who> [<access>] [<control>] ]+
    <what> ::= * | [dn[.<basic-style>]=<regex> | dn.<scope-style>=<DN>]
        [filter=<ldapfilter>] [attrs=<attrlist>]
    <basic-style> ::= regex | exact
    <scope-style> ::= base | one | subtree | children
    <attrlist> ::= <attr> [val[.<basic-style>]=<regex>] | <attr> , <attrlist>
    <attr> ::= <attrname> | entry | children
    <who> ::= * | [anonymous | users | self
            | dn[.<basic-style>]=<regex> | dn.<scope-style>=<DN>]
        [dnattr=<attrname>]
        [group[/<objectclass>[/<attrname>][.<basic-style>]]=<regex>]
        [peername[.<basic-style>]=<regex>]
        [sockname[.<basic-style>]=<regex>]
        [domain[.<basic-style>]=<regex>]
        [sockurl[.<basic-style>]=<regex>]
        [set=<setspec>]
        [aci=<attrname>]
    <access> ::= [self]{<level>|<priv>}
    <level> ::= none | disclose | auth | compare | search | read | write | manage
    <priv> ::= {=|+|-}{m|w|r|s|c|x|d|0}+
    <control> ::= [stop | continue | break]
```

指令中包含一个 to 语句，多个 by 语句。这个指令的大体意思是，通过 access to 约束我们访问的资源，通过 by 设定哪个用户（who）有什么权限，并控制（control）这个 by 语句完成后是否继续执行下一个 by 语句或者下一个 ACL 指令。

资源有多种形式，如 DN、属性、过滤条件。

（1）通过约束 DN 访问

例如：access to dn="uid=matt, ou=Users, dc=example, dc=com" by * none

这个指令是指访问 uid=matt, ou=Users, dc=example, dc=com 这个 DN，即把访问的范围约束在这个 DN 中。by * none 是指对于任何人的访问都是拒绝的。

总体的意思就是，任何人都没有权限访问 uid=matt, ou=Users, dc=example, dc=com 这个 DN。当然，服务器管理员是可以访问的，不然就无法维护这个 OpenLDAP 中的用户信息。

再例如：access to dn.subtree="ou=Users, dc=example, dc=com" by * none

在这个例子中使用了 dn.subtree。在我们的目录信息树中，在 ou=Users 子树下可能有