```
/* 打开连接 */
ld = ldap_open("dotted.host.name", LDAP_PORT);
if (ld == NULL )   exit( 1 );

/* 以 nobody 身份进行认证 */
if (ldap_simple_bind_s(ld, NULL, NULL ) != LDAP_SUCCESS )
{
    ldap_perror(ld, "ldap_simple_bind_s");
    exit(1);
}

/* 搜索 cn 为"Babs Jensen"的条目，返回所有属性 */
if (ldap_search_s(ld, "o=University of Michigan, c=US",
    LDAP_SCOPE_SUBTREE, "(cn=Babs Jensen)", NULL, 0, &res )
    != LDAP_SUCCESS )
{
    ldap_perror( ld, "ldap_search_s" );
    exit( 1 );
}

/* step through each entry returned */
for ( e = ldap_first_entry( ld, res ); e != NULL; e = ldap_next_entry( ld, e ) )
{
    /* print its name */
    dn = ldap_get_dn( ld, e );
    printf( "dn: %s0, dn );
    Ldap_memfree(dn);

    /* 输出每个属性 */
    for ( a = ldap_first_attribute( ld, e, &ptr ); a != NULL;
        a = ldap_next_attribute( ld, e, ptr ) )
    {
        printf( "attribute: %s0, a );
        /* 输出每个属性值 */
        vals = ldap_get_values( ld, e, a );
        for ( i = 0; vals[i] != NULL; i++ )
        {
            printf( "value: %s0, vals[i] );
        }
        ldap_value_free( vals );
    }
}
```

```
    /* 释放搜索结果集 */
    ldap_msgfree( res );

    /* 关闭和释放连接资源 */
    ldap_unbind( ld );
}
```

6.4.3　LDAP 应用案例

LDAP 的优势是查询速度快，以信息树的形式存储数据，LDAP 特别适合身份认证、资源管理、CA 等系统的数据管理和发布，这些系统的数据符合目录信息树的特点。例如，身份认证系统管理的是一个机构的实体信息，而机构组织结构本身就可以表示成一个信息树。在 CA 系统中，证书的颁发者和使用者就是使用与 LDAP 完全等同的 DN 表示的，用 LDAP 存储要发布的证书和 CRL 就非常合适。在 CA 系统中一般采用如图 6-9 所示的逻辑结构。

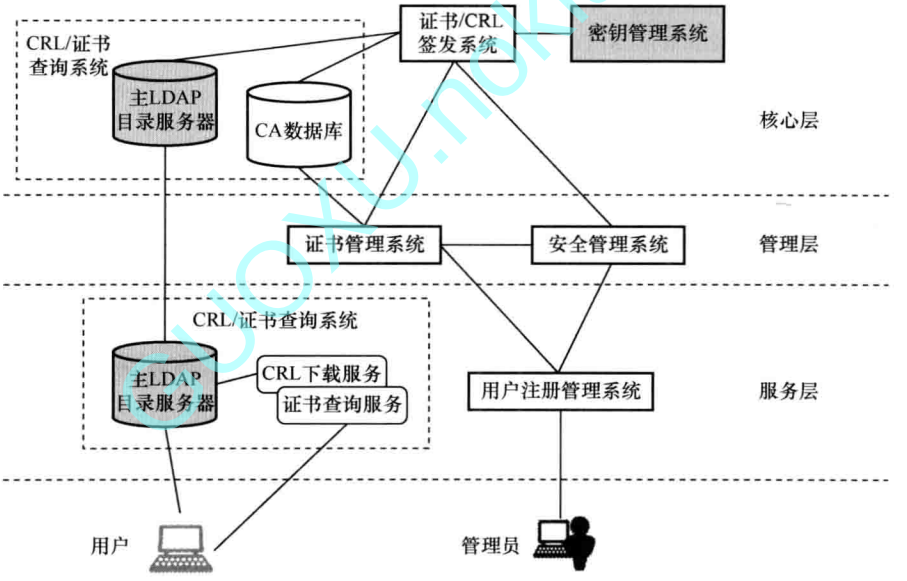

图 6-9　CA 系统逻辑结构

从图中可以看出 CA 系统分三层结构，分别是核心层、管理层、服务层，其中核心层的安全级别最高，主目录服务部署在核心层，由“证书/CRL 签发系统”向主目录写入已签发的证书和 CRL。而向用户提供的证书和 CRL 查询服务由从目录服务提供，从目录服务实时从主目录服务同步证书和 CRL。

第7章 实 验 一

7.1 DER 编码示例：X.501 Name 类型

7.1.1 ASN.1 描述与实例

1. ASN.1 描述

X.501 Name 类型用 ASN.1 描述如下：

```
Name ::= CHOICE {   RDNSequence   }
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
RelativeDistinguishedName ::= SET OF AttributeValueAssertion
AttributeValueAssertion ::= SEQUENCE {
    AttributeType,
    AttributeValue }
AttributeType ::= OBJECT IDENTIFIER
AttributeValue ::= ANY
```

Name 类型定义为 CHOICE 类型，目前只有 1 个选项 RDNSequence。RDNSequence 定义为 SEQUENCE OF 类型，由 0 个或多个 RelativeDistinguishedName 组成。RelativeDistinguishedName 定义为 SET OF 类型，由 0 个或多个 AttributeValueAssertion 组成。AttributeValueAssertion 定义为 SEQUENCE 类型，由 2 个成分组成：1 个为 AttributeType 类型和 1 个 AttributeValue 类型。AttributeType 定义为 OBJECT IDENTIFIER 类型。AttributeValue 定义为 ANY 类型，具体内容由 AttributeType 决定。

事实上，Name 类型可理解为分层或树形结构，即 X.500 目录树结构。

2. Name 实例

对于用户 Test User 1，其对应的 Name 类型采用分层结构描述为：

```
                (root)
                  |
        countryName = "US"
                  |
organization Name = "Example Organization"
                  |
      commonName = "Test User 1"
```

其中，每层对应一个 RelativeDistinguishedName；每个 RelativeDistinguishedName 由 1 个 AttributeValueAssertion 组成。等号前内容为 AttributeType，等号后内容为 AttributeValue。

用户 Test User 1 包含 3 个 AttributeType：countryName、organizationName、commonName，其 OID 定义如下：

```
attributeType OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) ds(5) 4 }
countryName OBJECT IDENTIFIER ::= { attributeType 6 }
organizationName OBJECT IDENTIFIER ::= { attributeType 10 }
commonName OBJECT IDENTIFIER ::= { attributeType 3 }
```

7.1.2　DER 编码过程

1．AttributeType 编码

AttributeType 为 OBJECT IDENTIFIER 基本类型，编码规则采用基本类型定长模式。

对于标识串，采用低标识编码方式，只需 1 个字节。OBJECT IDENTIFIER 的 tag 为 0x06；class 选择 universal，则位 8 和位 7 为 0，OBJECT IDENTIFIER 为基本类型，则位 6 为 0。因此，标识串=0x06。

对于长度串，采用短型编码方式，只需 1 个字节。

对于内容串，由 3 个字节组成。2.5.4.6 编码为 55 04 06，2.5.4.10 编码为 55 04 0A，2.5.4.3 编码为 55 04 03。

具体编码过程如表 7-1 所示。

表 7-1　AttributeType 编码过程

AttributeType	OID 定义	标　识　串	长　度　串	内　容　串
countryName	2.5.4.6	06	03	55 04 06
organizationName	2.5.4.10	06	03	55 04 0A
commonName	2.5.4.3	06	03	55 04 03

2．AttributeValue 编码

AttributeValue 为 PrintableString 基本类型，编码规则采用基本类型定长模式。

对于标识串，采用低标识编码方式，只需 1 个字节。PrintableString 的 tag 为 0x13；class 选择 universal，则位 8 和位 7 为 0，OBJECT IDENTIFIER 为基本类型，则位 6 为 0。因此，标识串=0x13。

对于长度串，采用短型编码方式，只需 1 个字节。

对于内容串，由其 ASCII 码组成。

具体编码过程如表 7-2 所示。

表 7-2　AttributeValue 编码过程

AttributeValue	标识串	长度串	内容串
"US"	13	02	55 53
"Example rganization"	13	14	45 78 61 6D 70 6C 65 20 4F 72 67 61 6E 69 7A 61 74 69 6F 6E
"Test User 1"	13	0B	54 65 73 74 20 55 73 65 72 20 31

3．AttributeValueAssertion 编码

AttributeValueAssertion 为 SEQUENCE 结构类型，编码规则采用结构类型定长模式。

对于标识串，采用低标识编码方式，只需 1 个字节。SEQUENCE 的 tag 为 0x10；class

选择 universal，则位 8 和位 7 为 0，SEQUENCE 为结构类型，则位 6 为 1。因此，标识串=0x30。

对于长度串，采用短型编码方式，只需 1 个字节。

对于内容串，由 AttributeType 和 AttributeValue 的 DER 编码值组成。

具体编码过程如表 7-3 所示。

表 7-3　AttributeValueAssertion 编码过程

AttributeValueAssertion	标　识　串	长　度　串	内　容　串
countryName ="US"	30	09	06 03 55 04 06 13 02 55 53
organizationName= "Example rganization"	30	1B	06 03 55 04 0A 13 14 45 78 ... 6F 6E
commonName = "Test User 1"	30	12	06 03 55 04 03 13 0B 54 65 ... 20 31

4．RelativeDistinguishedName 编码

RelativeDistinguishedName 为 SET OF 结构类型，编码规则采用结构类型定长模式。

对于标识串，采用低标识编码方式，只需 1 个字节。SET OF 的 tag 为 0x11；class 选择 universal，则位 8 和位 7 为 0，SET OF 为结构类型，则位 6 为 1。因此，标识串=0x31。

对于长度串，采用短型编码方式，只需 1 个字节。

对于内容串，由 AttributeValueAssertion 的 DER 编码值组成。

具体编码过程如表 7-4 所示。

表 7-4　RelativeDistinguishedName 编码过程

RelativeDistinguishedName	标　识　串	长　度　串	内　容　串
countryName ="US"	31	0B	30 09 06 03 55 04 06 13 02 55 53
organizationName= "Example rganization"	31	1D	30 1B 06 03 55 04 0A 13 14 45 78 ... 6F 6E
commonName = "Test User 1"	31	14	30 12 06 03 55 04 03 13 0B 54 65 ... 20 31

5．RDNSequence 编码

RDNSequence 为 SEQUENCE OF 结构类型，编码规则采用结构类型定长模式。

对于标识串，采用低标识编码方式，只需 1 个字节。SEQUENCE OF 的 tag 为 0x10；class 选择 universal，则位 8 和位 7 为 0，SEQUENCE OF 为结构类型，则位 6 为 1。因此，标识串=0x30。

对于长度串，采用短型编码方式，只需 1 个字节。

对于内容串，由 3 个 RelativeDistinguishedName 的 DER 编码值组成。

具体编码过程如表 7-5 所示。