



## Interaction of H.350 with client and Server

Nadim E. El-Khoury  
University of North Carolina at  
Chapel Hill

[Nadim\\_Elkhoury@unc.edu](mailto:Nadim_Elkhoury@unc.edu)



## Available SDKs, C and Java

- OpenLDAP <http://www.openldap.org>
- Mozilla  
<http://ftp.mozilla.org/pub/mozilla.org/directory/>
- Netscape  
<http://developer.netscape.com/tech/directory/downloads.html#source>
- LDAP Class Libraries Java by Novell  
<http://www.openldap.org/jldap>
- Java JDBC – LDAP Bridge Driver  
<http://www.openldap.org/jdbcldap/>



## SDKs Continued



- Sun's Java Naming and Directory Interfaces (JNDI)
  - Service providers like JDBC drivers
  - Support for LDAP, NIS, NDS, other providers



- Microsoft Active Directory Interfaces (ADSI)
  - Support for LDAP, NIS, NDS, ADS, other providers (e.g NT Domain)





## Connect to LDAP



- Multiple operation can share a single LDAP connection
- Standard port is 389 for non SSL
- Standard port is 636 for SSL





## Bind to LDAP Server

- Binding is basic LDAP authentication
- LDAP data security is based on how client is binded
  - Anonymous can search
  - Binded clients can modify
- Binding is separate from connection



## LDAP Bind



- Connect
- Authenticate with your id and password or certificate
  - Real bind with DN not just userid
  - Encryption dependent on client and server
- Perform operations
- Disconnect





```
#include <ldap.h>
#define LDAP_VERSION 3

LDAP *ld;
int rc;

/* get handle to an LDAP connection */
if ( (ld = ldap_init( ldap_host, LDAP_PORT )) == NULL)
{
    perror( "ldap_init");
    return (-1);
}

/* Sets the right version for the LDAP server */
rc = LDAP_VERSION3;
if( (ldap_set_option(ld, LDAP_OPT_PROTOCOL_VERSION, &rc ) )!=
LDAP_SUCCESS)
{
    ldap_perror (ld, "ldap_set_option" );
    return (-4);
}
```



```
/* Authenticates to the directory with user name and password */  
if (ldap_simple_bind_s (ld, user, password ) != LDAP_SUCCESS)  
{  
    ldap_perror (ld, "ldap_simple_bind_s" );  
    return (-2);  
}  
  
/* authenticate to the directory anonymously */  
if (ldap_simple_bind_s (ld, NULL, PWD ) != LDAP_SUCCESS)  
{  
    ldap_perror (ld, "ldap_simple_bind_s" );  
    return (-2);  
}
```



## Connecting over SSL

- Call `ldapssl_client_init()` if you don't plan on using certificate-based client authentication
- Call `ldapssl_clientauth_init()` if you plan certificate-based client authentication
- Connect to the LDAP server using `ldapssl_init()`



```
#include <ldap_ssl.h>
#include <stdio.h>

/* Initializes client, using our certificate database. */
if ( ldapssl_client_init( "/mypath/cert7.db", NULL ) < 0)
{
    printf( "Failed to initialize SSL client...\n" );
    return( 1 );
}
/* Gets a handle to an LDAP connection.*/
if ( (ld = ldapssl_init( ldap_host, LDAPS_PORT, 1 )) == NULL
{
    perror( "ldapssl_init" );
    return( 1 );
}
```



## LDAP URLs

- LDAP URL
  - ldap[s]://<hostname>:<port>/<base\_dn>?<attributes>?<scope>?<filter>
- ldap://videnet.unc.edu/ou=people,dc=vide,dc=net?commURI?sub?(&(sn=elkhoury)(commuri=\*))
- Makes LDAP accessible from browsers
  - Netscape 4 and higher
  - Mozilla
  - IE 5 and higher



## commURI

uid=elkhoury,ou=people,dc=vide,dc=net

objectclass: inetorgperson

objectclass: commURIObject

uid: elkhoury

commURI: ldap://videnet.unc.edu/dc=vide,dc=net??sub?(commUniqueId=54)

commURI: ldap://videnet.unc.edu/dc=vide,dc=net??sub?(commUniqueId=53)



commUniqueId=54,ou=h323identity,dc=vide,dc=net

objectclass: commObject

objectclass: h323Identity

commUniqueId: 54

commOwner: ldap://videnet.unc.edu/dc=vide,dc=net??sub?(uid=elkhoury)

h323Identityh323-ID: Nadim\_Elkhoury

h323IdentitydialedDigits: 0011297001



```
#define DEFAULT_FILTER "(objectclass=*)"

char DEFAULT_BASE [120] = "";
char *attrs[2];

/* Retrieve commURI */
attrs[0] = "commuri";
attrs[1] = NULL;

/* Build search base using the user id */
sprintf(DEFAULT_BASE, "uid=%s,ou=people,dc=vide,dc=net", user);

if (ldap_search_s(ld, DEFAULT_BASE,
                 LDAP_SCOPE_BASE, DEFAULT_FILTER, attrs, 0, &result) !=
    LDAP_SUCCESS)
{
    ldap_perror (ld, "ldap_search_s");
    return (1);
}
```



```
if ( (e = ldap_first_entry(ld, result)) != NULL)
{
    if ( (vals = ldap_get_values( ld, e, "commuri" )) != NULL)
    {
        for ( i = 0; vals[i] != NULL; i++)
        {
            url = vals[i];
            printf ("%s\n", vals[i]);

            /* Let's parse the URL */
            /* So we can search the specified server */
            if (( err = ldap_url_parse( url, &ludp )) != 0 )
            {
                fprintf( stderr, "ldap_url_parse: error %d\n", err );
            }
            else
            {
                printf( "\t host: " );
                if ( ludp->lud_host == NULL )
                {
                    printf( "DEFAULT\n" );
                    HOST = "localhost";
                }
            }
        }
    }
}
```



```
else
{
    printf( "<%s>\n", ludp->lud_host );
    HOST = ludp->lud_host;
}
printf( "\t port: " );
if ( ludp->lud_port == 0 )
{
    printf( "DEFAULT\n" );
    PORT = LDAP_PORT;
}
else
{
    printf( "%d\n", ludp->lud_port );
    PORT = ludp->lud_port;
}
SEARCHBASE = ludp->lud_dn;
printf( "\t dn: <%s>\n", ludp->lud_dn );
printf( "\t attrs:" );
if ( ludp->lud_attrs == NULL )
{
    printf( " ALL" );
}
}
```



else

```
{
  for ( i = 0; ludp->lud_attrs[ i ] != NULL; ++i )
  {
    printf( " <%s>", ludp->lud_attrs[ i ] );
  }
}
printf( "\n\t scope: %s\n", ludp->lud_scope ==
LDAP_SCOPE_ONELEVEL ?
    "ONE" : ludp->lud_scope == LDAP_SCOPE_BASE ?
"BASE" :
    ludp->lud_scope == LDAP_SCOPE_SUBTREE ? "SUB" :
    "**invalid**" );
printf( "\tfilter: <%s>\n\n", ludp->lud_filter );
```



```
/* Let's now call the function that will follow the LDAP URL and retrieve the */  
/* information. */  
  
rc = url_ldap_search (ludp);  
if (rc == -1 )  
{  
    printf ( " Could not Initialize our connection to the LDAP server: %s\n",HOST);  
}  
else if (rc == -2)  
{  
    printf ( "Failed to bind to the LDAP server: %s\n", HOST);  
}  
else if (rc == -3)  
{  
    printf ( "Failed to find any information for the following filter: %s\n",ludp->lud_filter);  
}  
else if (rc == -4)  
{  
    printf ( "We ran into an error in the ldap_result function\n");  
}  
  
ldap_free_urldesc( ludp );  
}
```



```
}  
    ldap_value_free (vals);  
}  
}  
ldap_msgfree(result);  
ldap_unbind(ld);
```



## LDAP Search (White Page lookup)



- PHP based code with embedded HTML code

