NORTHWESTERN
UNIVERSITY

Central Authentication Services Through LDAP

Revision 4
June 2007

Summary 2
Architectural Summary 2
Policy Considerations 2
Technical Specifics of Using LDAP for Authentication 3
Northwestern's central LDAP servers 3
LDAP White Pages 3
LDAP Registry 4
Authorities Providing Data 4
High Points of the LDAP Registry Schema 4
Connecting to the Central LDAP servers 5
How to Authenticate with LDAP 5
Use of SSL to protect connections 6
A Second-rate Way to Authenticate 6
Unsupported Methods to Authenticate 7
Use of LDAP Attributes in Software Applications 7
Application-driven Additions to the LDAP schema 7
Dos and Don'ts 8
Appendices 9
Appendix: A Short Technical Overview of LDAP 9
Our Directory is Flat 9
Standards for LDAP Attributes of People 10
The Syntax of a Distinguished Name 10
Binding 10
Appendix: White Pages Privacy Model 11
How to Proxy White Pages Information 12
What's an "On-Campus?" address 12
Appendix: LDAP Apache Authentication Example 13
Appendix: Registry Access Request Form 15
Appendix: References 16
University Policy References 16
Technical References 16

Summary

NU Information Technology (NUIT) is deploying LDAP ("Lightweight Directory Access Protocol") as the primary authentication service. Commercial or locally written applications of general use should require NetID and NetID-password authentication through the LDAP service. Applications that comply will be best positioned to use of future technologies deployed to enhance security. This document provides technical guidance to software programmers and service administrators to link applications to LDAP authentication.

NUIT is also clarifying policies on publishing information about people in the online directory "White Pages", and about release of information to application software. NUIT does not itself approve release of information to software applications, but acts as a clearinghouse for requests to the information custodians for access. The University units that are the sources of information, and which authorize the association of people with the University, will govern access to LDAP information.

Architectural Summary

The LDAP service is divided into two sub-services: the Registry and the White Pages. The White Pages service is the public online directory available to e-mail client software (Eudora, Outlook, etc.) and through the Web <http://directory.northwestern.edu>. Privacy of individuals and data items in the White Pages service is covered under "White Pages Privacy Model" below.

The LDAP Registry is the comprehensive database of identity, password, attribute, and other information relevant for applications. This service contains all information within the White Pages service plus extensive role and entitlement indicators.

The Registry service is populated by the SNAP meta-directory system, which receives authoritative identity information from systems such as Student Records and Human Resources. The SNAP system combines information from these sources, applies business rules about permissions, privacy, etc., and delivers information to the Registry. The Registry then passes only the necessary information to the White Pages service.

Policy Considerations

Groups developing information systems using NetID authentication, or LDAP directory information, must make a written request for the information they need. NUIT grants access to authentication services; however, the University service units providing authoritative information, such as Human Resources and the Registrar, govern release of specific LDAP attributes to software applications.

Other policies about privacy and information use regarding various groups of people, (such as applicants, registered students, alumni, or faculty), are developed in consultation with the central units authorizing the existence of University identities for those groups of people. (For example, Human Resources, sets privacy policies for staff.) When access is granted, there likely will be accompanying policy constraints on how the data items may be viewed, searched, or locally

stored by the application. The LDAP schema includes fields that specify more fine-grained control over privacy than was previously available. It is the responsibility of application developers to implement control of the release of this information by their applications, consistent with these fields.

When possible, refer directly to centrally services that publish information. For example, use "directory.northwestern.edu" as a White Pages service, and use "search.northwestern.edu", as a web search engine. If it is necessary to release White Pages directory data by proxy, then we recommend specific technical means to do this.

In planning for new software systems, LDAP is the preferred technical mechanism to specify for authentication, and access to university data.

Technical Specifics of Using LDAP for Authentication

What follows is a technical description of how to use Northwestern's central LDAP servers in software applications. Those not familiar with LDAP may wish to read the appendix: "A Short Technical Overview of LDAP."

Northwestern's central LDAP servers

The Registry LDAP host name is "registry.northwestern.edu".

The White Pages host name is "directory.northwestern.edu".

The search base for all NUIT LDAP servers is: "dc=northwestern,dc=edu"

LDAP White Pages

The White Pages service exists to provide public directory information to the online directory web page, and e-mail/web client software (Eudora, Outlook, etc.), which query LDAP directories directly. The White Pages service allows anonymous binds from on and off-campus on the default LDAP port, port 389. The host name for the service is "directory.northwestern.edu".

The user controls how much information is returned to queries from off-campus. In general, off-campus queries will receive less information than requests coming from on-campus.

The online directory web pages at <http://directory.northwestern.edu> are the primary user interface to the campus online directory. There is also an authenticated web page that provides an interface for making NetID-authenticated requests from off-campus. This provides the equivalent of on-campus access for someone with a valid NetID.

Certain fields in the White Pages service are available for the user to populate and edit through specific Web pages, as are also privacy settings. Changing information from an authoritative source must take place within that source's database, not separately in the online directory.

LDAP Registry

The Registry service is a data repository for the use of University software applications. Connection the Registry service requires a service DN/password and is also restricted by IP address and/or network. The preferred way to connect with the Registry is to use SSL on port 636 and do a "simple" bind with a DN and password. The host name for the service is "registry.northwestern.edu". **See important information on temporary host names above.**

The Registry servers have more attributes available than are exposed in the directory servers. Application service DNs will be given access to fields based on the needs of the application.

Access to the Registry is IP-restricted. An application for LDAP access should include IP addresses to be used in testing in production.

Authorities Providing Data

The majority of LDAP directory entries for people come from nightly feeds of data provided from the Human Resources and Student Records system. These are read and processed by the "SNAP" meta-directory software, which moves identity information to various services and applies local policies.

Additional directory data is supplied for organizations and alumni from information entered by the NUIT staff responsible for creating special accounts manually and staff in various schools who are authorized to add or edit some groups of NetIDs.

High Points of the LDAP Registry Schema

The LDAP Registry schema is at <http://www.it.northwestern.edu/isa/registry/schema.html>.

NUIT has included some attributes from the standard object classes organizationalPerson, inetOrgPerson, and eduPerson. Those attributes that are local additions have names prefixed with "nu". Some particular fields of note are:

- *uid*. The NetID is the single-valued "unique identifier".
- *cn*. The "common name" is a multi-valued field searched by most e-mail software and must contain combinations of first name, last name, and nicknames. Most e-mail software allows this field to be hidden when displaying a search result to the user.
- *displayName*. The field "displayName" is singled-valued and holds the preferred form of a person's name for display.
- *mail*. The field "mail" is single-valued and holds the preferred form of a person's e-mail address. This will be the single e-mail address published for this person.
- *numail*. The field "numail" is multi-valued and represents all valid e-mail addresses for this person, taking into account special domains.
- *nuIdTag*. The field "nuIdTag" is multi-valued and contains all valid identifier tokens for this person. At a minimum, this field contains the NetID. It may contain other values based upon source of the identity.

Connecting to the Central LDAP servers

At this time, for access to the central White Pages and Registry LDAP servers, three means of access are supported:

- Anonymous bind (without a DN or credentials) — applies to White Pages only.
- A “simple” bind using a DN and password sent in clear text on the default LDAP port.
- A “simple” bind using a DN and password sent on the alternate LDAP port 636, with SSL encryption protecting the session.

Microsoft Active Directory uses additional types of LDAP authentication, including some based on a form of Kerberos 5, but this is not implemented on the primary campus servers, and NUIT is moving away from the use of Kerberos in applications.

There is a extension framework for various kinds of authentication called SASL (Simple Authentication and Security Layer), used with the LDAP protocols, which defines a number of other kinds of authentication for binding to LDAP. At this time, none of the SASL authentication methods is implemented on the NUIT White Pages or Registry servers. With some software toolkits, in some contexts, it may be necessary to turn off SASL negotiation to do just a “simple” bind to the NUIT LDAP servers.

NUIT is using the SUN Java Enterprise Directory server product for the LDAP servers. Most open-source LDAP projects are based partly on the OpenLDAP project and the OpenSSL cryptography toolkit. Microsoft Active Directory includes its own versions of LDAP software. SUN, Microsoft, and OpenLDAP LDAP clients and servers differ in details of implementation, and features, but interoperate to the extent that they are based on common standards.

How to Authenticate with LDAP

Authenticating against LDAP with a NetID and password is really checking the password associated with a particular Distinguished Name. The preferred way to do LDAP authentication in an application is a two-stage process, where the software binds, searches, and binds again.

A good example of this is the pair of LDAP modules in the Apache 2 distribution mod_ldap and mod_auth_ldap. See the appendix: “Apache Authentication Example.”

For applications needing to authenticate against LDAP or use LDAP attributes, NUIT will issue an individual “service” DN and password allowing access to the Registry LDAP servers. This access will be restricted further by IP address, and limited to specific fields in the LDAP database.

The algorithm to check a username/password pair is:

1. Bind to a Registry server with the service DN and service password
2. Search, from the base, “dc=northwestern, dc=edu”, using subtree scope, for the DN of a directory entry which matches some LDAP attribute with the username provided.

   a. The preferred field for this search is "*nuIdTag*"
   b. This search can optionally include an LDAP search filter and/or retrieve other attributes visible to the service DN that the application needs, such as uid
3. Unless the search matches exactly one DN, authentication fails
4. Bind to the LDAP server using the DN returned by the search and the password provided
5. If this second bind succeeds, authentication succeeds, otherwise it fails.

This approach is recommended because it does not require making too many assumptions about how the LDAP directory is structured, or how passwords are stored in LDAP.

If you search on "*nuIdTag*", your software needs to be prepared to accept long "username" values, including the characters allowed in an RFC822 e-mail address without embedded comments. If your application needs to know the real NetID, it should also get uid as an additional attribute while searching for the DN, or do a subsequent search for uid while bound as the service DN.

The motivation for using "*nuIdTag*" is to support logins in cases where someone may not know their actual NetID because it is generated behind the scenes.

Use of SSL to protect connections

Used by itself, the LDAP protocol sends information in the clear, with no encryption. Integrity and confidentiality can be protected by using SSL as a transport. In production use, all LDAP connections should be done with SSL to port 636. The only exceptions that may be made to this rule are applications using only highly secure networks, hosted by within the University Data Center. Even then, the best-practice is to use SSL to keep passwords off the network.

In software development, it may be necessary to do tests with and without SSL, because errors in SSL set-up can produce cryptic errors, indistinguishable on the client side from several other errors. It will be necessary to configure SSL client software to trust the SSL certificate on the central LDAP servers. This could be done by configuring trust for NUIT's specific certificate/certificate authority, or trusting a "laundry-list" of well-known certificate authorities. The Apache authentication example contains a file "verisignca.pem" which can be used to set up trust for the NUIT certificate authority in Apache.

In addition to protecting LDAP queries, when usernames and passwords are being used in web authentication of web requests, web traffic should also be protected with https/SSL.

A Second-rate Way to Authenticate

A technically inferior way to do authentication against LDAP involves building into the application the knowledge of how to map a username to a DN, and using that mapping to replace the initial bind and search for a DN. In this case, the mapping would be from username to the DN:

uid=username, ou=People, dc=northwestern, dc=edu

This method is depreciated, because it is more sensitive to changes in the LDAP schema and undermines the University's ability to deploy other identifiers in special circumstances.

Unsupported Methods to Authenticate

Some software's notion of authenticating with LDAP is to retrieve the encrypted userPassword hash from the LDAP directory and to compare the salt and hash with the supplied password directly. This will not be allowed: applications will not be granted access to see the password hashes.

Use of LDAP Attributes in Software Applications

Software projects that need information beyond just NetID authentication should detail their requirements in the request for access to LDAP. As noted above, the initial search for a DN can be used return other attributes visible to a service DN, beyond the user's DN. These may be used in an LDAP search filter, or processed with other logic in the application.

Since service DNs connected to the Registry can potentially see information not exposed in the White Pages, applications should be designed to use the Registry directly, not to look up people in the White Pages directory. The exception to this would be when a "censored" view similar to what's seen in White Pages is desired in some part of application; in that case it is still necessary to consider the White Pages privacy rules.

Application-driven Additions to the LDAP schema

If a university unit wants to expose information from LDAP for which they will be the authoritative source, NUIT will consider requests to add new LDAP attributes to the Registry schema. Application software cannot be granted the ability to change the LDAP schema themselves. Exposing attributes in the LDAP Registry is particularly suited to allowing other departments to make authorization decisions based on your information, after authenticating a NetID against LDAP.

There are security and performance trade-offs for using the LDAP Registry to house data items versus exposing items directly from a database table. Compared to a database, LDAP servers are better suited to referencing moderate volumes of data that is searched or fetched frequently but modified infrequently. Due to wide deployment of LDAP, the Registry may serve as a common denominator between dissimilar or loosely coupled systems. Other LDAP servers, or other software mechanisms, could enforce security policies not offered by the central NUIT Registry servers. Units and developers should consult with NUIT early in any process where local LDAP services are anticipated.

Dos and Don’ts

- Don’t:
  - Don’t assume that the DN can be derived from the token passed to the application as the “NetID”. Search the “*nuIdTag*” field for the token and retrieve the DN of the result and the *uid* field.
  - Don’t do LDAP NetID authentication against Active Directory. NetID authentication against Active Directory using LDAP is feasible, but is not recommended for software products that do not already have a natural relationship to Active Directory.
  - Don’t use Kerberos in new applications. LDAP authentication using SSL is preferred for new applications over Kerberos. The exception would be the native use of Kerberos in Windows/Active Directory.
- Do:
  - Do search the “*nuIdTag*” field for match on the NetID and retrieve the *uid*.

Appendices

Appendix: A Short Technical Overview of LDAP

This appendix is a quick overview of the characteristics of LDAP that are relevant to configuring software to use the NUIT services. It is written for people already familiar with computers and Internet protocols, but with relatively little knowledge of LDAP.

LDAP is a directory service protocol developed by a working group of the IETF (Internet Engineering Task Force). It was originally based on another international standard, the X.500 ITU (International Telecommunications Union) series of specifications. As such, LDAP uses a mix of terms and technology from both X.500, and other Internet protocols. This results in novel syntax and naming conventions being used for otherwise familiar concepts.

The protocols and their syntax are formally expressed in ASN.1 (Abstract Syntax Notation One). But as a first approximation, one can ignore this, and just treat directory entries as a bunch of strings with some attached data types and constraints.

An LDAP directory is a tree structure of objects. Each object has a data type and a list of attributes (with associated data types). Any object has the potential to be a container, with child objects under it in the tree structure. There are predefined objects, defined in several standards, which are intended to structure a directory tree for an entire organization. LDAP uses an adaptation of Internet Domain Name Service (DNS) names to provide a global frame of reference, in addition to the global hierarchy envisioned by X.500.

The administrators of a LDAP directory server control what object types and data syntax rules are allowed on a server. This is collectively known as the server's "schema". LDAP attributes can be single-valued or multi-valued. If an attribute is multi-valued, there is no intrinsic order attached to those values, though order is sometimes preserved. LDAP attributes have attached syntax rules that determine what are allowed values, and whether or not values are case-sensitive, or even have binary values.

Our Directory is Flat

Potentially, we could build LDAP directory entries for people in a tree structure that mirrored the organization chart in great detail. In practice, for institutions like ours, this has turned out to be more trouble than it is worth, because of people moving within the organization and between roles.

In our case, the directory entries for people, (and organizational NetIDs), are all located in a particular place in the directory tree. Each directory entry is a flat list of attributes with no child objects below it.

Standards for LDAP Attributes of People

There is a series of standards that have defined more attributes and syntax rules to define what is a "person":

- The object class "*organizationalPerson*" was defined in by the ITU in X.521 (2001)
- The object class "*inetOrgPerson*" was defined by the IETF in RFC 2798
- The object class "*eduPerson*" was defined by Internet2/EDUCAUSE in specifications published at: <http://www.educause.edu/eduperson/>

NUIT is including attributes from these standards to allow for future interoperability with our educational peer institutions.

The Syntax of a Distinguished Name

The path to an LDAP directory entry isn't expressed like a URL or a Windows or Unix file path, but as a list of attributes, going down the directory tree. This path to a directory entry is called a "Distinguished Name" or "DN". There may be more than one equivalent way to write a DN that refers to the same directory entry, but a DN must refer to a single directory entry. Usually, in a given context, DNs are displayed in some stereotyped form. For, example, within the Northwestern LDAP schema, a person with NetID "johndoe" would have the DN:

```
uid=johndoe, ou=People, dc=northwestern, dc=edu
```

The "ou" and "dc" attributes aren't case-sensitive and the white space after the commas is optional. The DNs for the university as a whole start at the "base" DN of:

```
dc=northwestern,dc=edu
```

Binding

LDAP directory services are designed to give out some information anonymously, and restrict other information only to an identity that has authenticated to the server. Setting up a session to an LDAP server is called "binding". If the LDAP server permits applications to bind without authenticating themselves, then the bind is termed "anonymous". This is the type of bind used by e-mail client software (Eudora, Outlook, etc.) to search the White Pages service.

The NUIT LDAP Registry service does not allow anonymous binds. All access to services of the Registry requires that an application bind as itself (with an application-specific DN and password) and then make queries against the database. This is an important security feature because each binding DN is assigned a view of the database, called an access control list (ACL), which defines which data items the application can see, change, or search upon. There is a separate ACL that is assigned to anonymous binds.

Appendix: White Pages Privacy Model

This is a summary intended to document the operational logic used to determine, based on fields in LDAP, what is allowed to be visible to white-pages queries. It's not a complete description of the policies which set up the values of those fields. More details are given in LDAP Registry schema documentation Web site <http://www.it.northwestern.edu/isa/registry/schema.html> This information is provided mainly to motivate an understanding of the LDAP schema design.

Software application designers should make explicit in your request for access to LDAP attributes, what information your software would expose, to whom. Your agreement with the data providing authority will govern what can be done with the data.

As a general principle, applications shouldn't expose more information to the public than is shown in a white-pages query, and should only use or show what is necessary.

We don't encourage software developers to *directly* implement their in applications all the privacy-related logic that follows, because the details are complicated, and may change over time. Instead, see the next section: "How To Proxy White Pages Information", for a mechanism to reproduce the same access control rules in a simpler way. (But your applications must respect these rules *somehow*.)

The primary factors affecting what is visible in the White Pages are:

- Whether the directory entry represents a student, faculty, or staff person.
- Whether the person has requested to be "Unlisted" ("*nuUnlisted*") set true
- What view is requested for off-campus queries ("*nuOffCampus*") set "full", "none", or "partial"
- What items are listed in "*nuPrivacyRestrictedFields*"
- What items listed in "*nuPartialResponseData*"

In addition, the data feed from Human Resources contains as many as 10 positions, with an associated block of data and a flag for each one indicating if they are to be visible. Multi-valued fields are present in the Registry for this information: one containing information only for the "visible" positions, the other containing all positions, for example, title and "*nuAllTitle*"; or "*nuDepartmentTitle*" and "*nuAllDepartmentTitle*".

If the person is unlisted, White Pages queries return the same result as if they didn't exist. However, application software is still expected to authenticate them and treat them like real persons, but information about them must be hidden from public view.

The "*nuOffCampus*" field determines what is displayed for White Pages requests from unauthenticated off-campus web or LDAP queries:

- "full" means the same fields are displayed as in an on-campus query
- "partial" means the fields listed in "*nuPartialResponseData*" are returned
- "none" causes an off-campus query to be handled as if the person were unlisted

Note that requests authenticated with Northwestern NetIDs and passwords are treated like "on-campus" requests.

The field "*nuPrivacyRestrictedFields*" is a list of other fields that are not to be returned to directory query - even from on on-campus.

How to Proxy White Pages Information

It is preferable to point people directly at the web interface<http://directory.northwestern.edu> for directory information. However, it may be that some applications, such as an application portal, want to display white-pages information to the public in a different context.

This section describes how to proxy white-pages queries though an application without re-implementing all the access control logic.

Requests are to be treated as "on-campus" if the original request came from an "on-campus" IP address *or* the requestor has been authenticated as having a Northwestern NetID. An anonymous bind to the white-pages server, "directory.northwestern.edu", from a Northwestern address will return the "on-campus" form of the white pages information.

To get the same reduced view as an "off-campus" white pages query, connect and bind with LDAP to "directory.northwestern.edu" with the DN:

```
cn=offcampus, ou=outside, dc=northwestern, dc=edu
```

and the password: "offcampus".

What's an "On-Campus?" address

Simple applications can define an "on-campus" IP address as coming from an IP address, whose reverse DNS looks up to "northwestern.edu" and whose DNS lookup is round-trip consistent. (No one should be using an IP address on Northwestern networks without an assigned, consistent DNS address, but it does happen. The Apache web server, among others, checks for round-trip consistent DNS, on directories restricted by DNS. Also, PARANOID deny checks in tcp_wrappers exclude inconsistent DNS.)

It is also possible to test that the IP address of a request belongs to a list of Northwestern net-blocks. This will allow access from IP addresses with incorrect DNS. This is more difficult for a system administrators to maintain, and so is recommended as an option mainly for large services hosted and managed by ITCS.

Appendix: LDAP Apache Authentication Example

This document, and a library of example code files, provide an example of how to do Apache authentication against the NU LDAP Registry. The code files are located online at <http://www.it.northwestern.edu/isa/specifications/ldap-authentication/apache-sample-files.zip>

The example was tested on Red Hat 9.0, with OpenLDAP and OpenSSL as provided by Red Hat (with current patches), and Apache 2.x built from source. The two Apache modules used for authentication are LDAP modules bundled with the Apache 2.x distribution as experimental modules (which must be explicitly enabled when Apache is built.)

mod_ldap manages connections to the LDAP server and provides a result cache, for performance.

mod_auth_ldap does the actual authentication against LDAP.

Since NetID passwords should be protected in transit, *mod_ssl* should be used for https: on the end-user side.)

The Apache modules will build against any of several LDAP Software Development Kits. What is available will probably depend on the OS. OpenLDAP is the common choice for Linux and MacOSX, Sun ships its own LDAP SDK. When you start the server, the error log contains a note as to which SDK was used and if SSL support is available in mod_ldap:

```
[Tue Oct 14 18:09:20 2003] [notice] Apache/2.0.47 (Unix) configured -- resuming normal operations
[Tue Oct 14 18:09:20 2003] [info] Server built: Oct 14 2003 17:15:38
[Tue Oct 14 18:10:42 2003] [notice] SIGHUP received. Attempting to restart
[Tue Oct 14 18:10:42 2003] [notice] LDAP: Built with OpenLDAP LDAP SDK
[Tue Oct 14 18:10:42 2003] [notice] LDAP: SSL support available
```

The code example file “httpd-conf-extracts.txt” is quotes from the httpd.conf used for testing with added comments.

The code example file “my.build.simple” is a shell script used to run configure to build Apache for testing. This is a stripped down test, which tried to stay close to a default install, to illustrate just LDAP authentication and a few features useful for debugging.

These probably the most important options:

```
--enable-so \
--enable-ssl=shared \
--enable-auth-ldap=shared \
--enable-ldap=shared \
--with-ldap \
```

Generally, any external libraries like OpenSSL and OpenLDAP, which are linked into Apache modules need to be built with the same compiler and similar compiler options as the Apache module. This wasn’t a problem on Red Hat Linux, where the defaults just worked. It could be an

issue on environments like Solaris where both a vendor-provided compiler and gcc are in common use.

NUIT will assign a service DN and password for each application. This will have enough access to check passwords, and optionally access other attributes needed by a specific application.

The request for access must include what IP addresses you will be using to access the Registry server for development and production use. Access to the Registry server is IP-restricted.

The DN, Distinguished Name, is the path to an entry in the LDAP Registry. User NetIDs currently have a DN of the form:

```
uid=netid, ou=People, dc=northwestern, dc=edu
```

The module mod_auth_ldap operates by first binding as the service DN, searching for the DN of the user by some attribute such as "*nuIdTag*". It fails if it doesn't find exactly one matching entry in LDAP. Second, it binds as the user's DN and password to check the password. Optionally, one can specify an LDAP search filter to restrict access further, based on LDAP attributes. The example shows a group restriction based on a search filter matching eduPersonAffiliation, which is a standard multi-valued attribute.

Setting up LDAP with SSL requires specifying trusted certificate authorities. The example files include the specific Verisign certificate information for the current Registry server. You should also be able to use the certificate bundle file used for a client-side mod_ssl configuration.

This is an error log message from a (normal) failed authentication:

```
Tue Oct 14 18:10:56 2003] [warn] [client 129.105.188.80] [16399] auth_ldap authenticate: user lunde
authentication failed; URI /ldap-reg-auth/test.html [User not found][No such object], referer:
http://socrates.tss.northwestern.edu/ldap-reg-auth/
```

This is an error log message due to an error setting up the trusted certificate authority for SSL:

```
[Tue Oct 14 18:06:16 2003] [warn] [client 129.105.188.80] [4499] auth_ldap authenticate: user lunde
authentication failed; URI /ldap-reg-auth/ [LDAP: ldap_simple_bind_s() failed][Can't contact LDAP
server]
```

Unfortunately, several other mistakes, such as a bad URL for the LDAP server produce the same message as above.

For debugging, it may be useful to try to bind to your service DN from the command line with the "ldapsearch" command. For example (as one command line):

```
ldapsearch -h ldap2.itcs.northwestern.edu -D
'cn=surveyuser,ou=pwcheck,dc=northwestern,dc=edu' -w 'XXXXXX'
-b 'dc=northwestern,dc=edu' 'uid=lunde'
```

If you are testing with version of "ldapsearch" that comes with OpenLDAP, use the "-x" option to suppress SASL negotiation. This doesn't seem to be an issue on Solaris.

Appendix: Registry Access Request Form

To request access to LDAP authentication and/or attributes for use in application software, fill out the Registry Access Request Form at: <http://www.it.northwestern.edu/isa/specifications/ldap-authentication/instructions.rtf> with Microsoft Word and send it to: nuit-isa@northwestern.edu

Appendix: References

University Policy References

"Standards for Business Conduct":
(The section "Confidentiality" has general statements on the release of information)
http://www.northwestern.edu/hr/policiesweb/standards.pdf

"FERPA: Students Right to Access Records Release of Student Information Policy":
http://www.registrar.northwestern.edu/ferpa/

Technical References

OpenLDAP documents:
http://www.openldap.org/doc/

What is A Directory Service, What is LDAP:
http://www.openldap.org/doc/admin21/intro.html#What%20is%20a%20directory%20service

The Internet2 Middleware Group:
http://middleware.internet2.edu/

MACE: Middleware Architecture Committee for Education:
http://middleware.internet2.edu/MACE/

The eduPerson Object Class specification:
http://www.educause.edu/eduperson/

OpenLDAP FAQ-O-Matic:
http://www.openldap.org/faq/data/cache/1.html

A Recipe for Configuring and Operating LDAP Directories
by Michael R Gettes, Georgetown University:
http://www.duke.edu/~gettes/giia/ldap-recipe/index.htm

Sun Java Enterprise System for Education
(includes "Sun Java System Directory Server 5.2", Sun's latest LDAP directory server offering.):
http://www.sun.com/products-n-solutions/edu/promotions/javaenterprisesystem.html

IETF LDAP v3 working group charter:
http://www.ietf.org/html.charters/ldapbis-charter.html