

Extending Internet Services Via LDAP

James E. Dutton
Southern Illinois University
jimd@siu.edu

June 21, 2000

Abstract

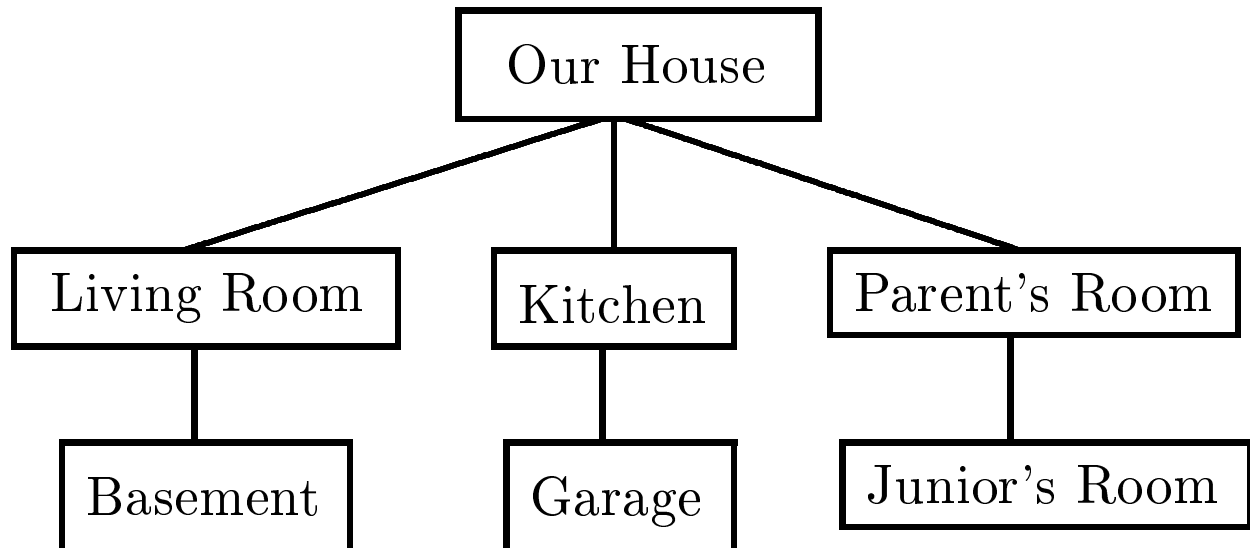
Using LDAP with INN, Sendmail, and Perl/REXX/JavaScript for special application controls and specialized LDAP queries with simple line-mode or nicer HTML formatted output.

2000 Usenix Technical Conference

LDAP Introduction

- LDAP can provide centralized datastore
- LDAP is easily user-extensible
- LDAP (line-mode) tools are easily accessible by scripting tools
- Combination of LDAP and application scripts can extend useability of application without major rewrite
- LDAP “fits” many platforms easily and inexpensively

Example of a simple LDAP database.



- LDAP does not require hierarchical structure, though data relationships may require or be better served by one
- in many cases, rearranging or adding to existing structures is easy
- structure may have impact on data access

LDAP attributes are generally simple text strings:

```
dn: cn=Living Room, ou=Our House
objectclass: room
description: where we gather together
size: 100ft x 200ft
wallcolor: O.D. Green
location: southwest corner
furniture: wide screen TV
furniture: fourteen chairs
furniture: three couches
furniture: five tables
furniture: credit card pay telephone
windowDressing: rainbow colored drapes
```

[P1] QLANADMIN Replaces Web Pages

Prior to LDAP deployment, local LAN/DNS Administrator information was made available via a Web page using a list format such as:

- Department #1
 - Subdomain(s): <subdomain>, <subdomain>
 - * Dept Unit(s): all
 - * Subnet(s): x1.000(TR), x1.128(E)
 - * <LAN Admin name>
 - <phone number>
 - <e-mail address>
- Department #2 . . .

(Line-mode) QLANADMIN is intended to provide simple and easy access to LAN Admin information without requiring a Web browser and searching a long Web page.

- Uses LDAPSEARCH command from LDAP distribution to perform LDAP searches
- Displays essential information in simple, but pleasant, line-mode
- Easy and quick to develop; short learning curve
- script written in both Perl and REXX

QLANADMIN (line-mode)

Sample Display

qlanadmin xxx.000

```
=====
Subdomain           : grdsch
Network Protocol    : IP
LAN Administrator:  FirstName LastName
Department          : Department Name
E-mail Address      : userid@mail.host
Telephone Number    : (xxx) xxx-xxxx
DNS Administrator:  FirstName LastName
=====
```

Sample QLANADMIN LDAP

dc=<subdom>,dc=<org>,dc=<orgtype>,o=<org>,...
objectclass=top
objectclass=domain
objectclass=localadmin
dc=<subdom>
description=Internet subdomain for <dept>
l=<campus location>
o=<subdom>
postofficebox=<mailbox>@<mailhost>
telephonenumber=<telephone number>
subdomain=<subdom.org.orgtype>
subnet=x1.000
subnet=x1.128
networktype=x1.000(TR), x1.128(E)
lanadmin=<LAN Admin DN>
dnsadmin=<DNS Admin DN>

QLANADMIN (web-mode)

Two web-based versions of QLANADMIN were developed that use multiple fixed options to select the LAN Admin information to search for.

- “static Web-mode display”
 - did not filter out any data returned
 - display results similar to [P2] “raw” display
- “dynamic Web-mode display”
 - uses JavaScript to control data returned and format of data displayed
 - provides additional features such as printing of data
 - is more complicated to develop

Dynamic Web-mode QLANADMIN Display

Close window

QLANadmin Results

Print window

Mon Apr03 17:43:47 GMT-0500(CDT)2000

Description	Internet subdomain for <department>
Network Name	subdomain.org.orgtype
Location	Woody Hall
E-mail Address	mailbox@mailhost
Telephone Number	<telephone number>
IP Router Subnet	xx1.000
IP Router Subnet	xx2.000
IP Router Subnet	xx3.000
IP Router Subnet	xx4.000
Subnet Network Type	xx1.000(TR), xx2.000(E), xx3.000(E), xx4.000(E)
LAN Admin	<name>
DNS Admin	<name>

JavaScript used to dynamically build HTML code to provide a specialized display. LDAP filter used to limit data returned.

[P2] Simple LDAP Query Using JavaScript

A simple Web page with a single input field to identify a specific LAN host to search for, uses JavaScript to create an “LDAP URL” and then open a new Web page with the results of the LDAP search.

- all data available for the specific target is returned; QLANADMIN filtered out some data
- no formatting of data returned is performed resulting in a “raw” format
- is very easy to develop
- Can provide LDAP access to user with no access to LDAP distribution and without special JavaScript requirements

Static Web-mode Display

Object Class	top
	domain
	localadmin
dc	ournet
Notes	Internet network domain..
networkname	our.network
City	location
Organization	department name
postofficebox	mailbox@mail.host
subdomain	sub.domain.name
subnet	999.999
dnsadmin	<LDAP DN>
lanadmin	<LDAP DN>
creatorsname	cn=ds_mgr
modifiersname	cn=ds_mgr

[P3] Sendmail Enhancements

- Project began in 1997/98 with Sendmail V8.8.8; current Sendmail-8.10.x has made changes with regards to LDAP
- Three sample uses of LDAP + Sendmail are implemented:
 - recognize “dotted” userids as special “local” users via LDAP
 - recognize .mlst mailbox suffixes as special “local” mailing lists via MAIL500
 - treat unresolved “local” userids as “local” via LDAP

Sample Sendmail Coding

Five basic additions are made to the `sendmail.cf` configuration file:

1. `DLuser ($L (USER_RELAY)` macro to force lookup of “local” userids in server password file)
2. Sendmail LDAP “map” declaration. The LDAP search key and attribute to return are specified with the `-k` and `-v` parms:

```
Kluser ldap
      -h"<LDAP server>"
      -k"cn=%s" -vmail
      -b"o=<orgname>,c=US"
```

-
3. in Ruleset 5, the “send unrecognized local users to a relay host” rule is activated and the following rule is added following it, which tells Sendmail to perform an LDAP search via the LDAP user “map” if the “local” userid being tested is not found

```
R< $L > $+  
    $: < $L > $(luser $1 $)
```

4. in Ruleset 98, two rules are added: one for the “local” mailing lists, the other for the dotted “local” userids:

```
# look for any (rfc822mailgroup)  
# <listname>.mlist, via Mail500  
R$- . mlist <@ $=w .>  
    $* $#mail500 @$2 $: <$1>
```

```
# anything like <firstname>.<lastname>
# at <this host> send to MAIL500
R$- . $- <@ $=w .>
    $* $#mail500 @$3 $: <$1.$2>
```

5. lastly, the MAIL500 definition which specifies the location of the executable

```
Mmail500,
    P=/usr/local/libexec/mail500, F=...,
    A=mail500 -f $f -h $h -m $n@$w $u
```

Sample SENDMAIL #1 LDAP

dn: ou=Test Mailing List, ou=Groups, ...
objectclass: rfc822mailgroup
objectclass: pilotObject
objectclass: localExtra
cn: Test Mailing List
cn: Test_ML
cn: Test.ML
description: test LDAP/MAIL500 (RFC822) mailing list
requeststo: cn=Mail Manager,ou=People,cn=Amiga1, ...
errorsto: cn=Mail Manager,ou=People,cn=Amiga1, ...
rfc822errorsto: jimd@amiga1.jim.dutton
rfc822requeststo: jimd@amiga1.jim.dutton
owner: cn=Mail Manager,ou=People, ...
mail: sloppy@freebsd1.jim.dutton
mail: newsadmin@amiga2.jim.dutton
member: cn=System Administrator,ou=People,...
member: cn=System Administrator,ou=People,...
member: cn=System Administrator,ou=People,...

Sample SENDMAIL #2 LDAP

dn: cn=<User Name>,ou=<Dept Name>, ...

objectclass: umichPerson

objectclass: person

description: User on host XYZ

cn: <User Name>

cn: <User_Name>

cn: <User.Name>

mail: <mailbox>@<mailhost>

othermailbox:<mailbox>@<other mailhost>

telephonenumber: <you can't afford it>

uid: <whatever>

userpassword: <tres chic>

[P4] Enhancing INN With LDAP

Basic newsgroup access control for InterNetwork News (INN) is:

- based upon host name/address, user authentication, user access permissions, and newsgroup list
- adequate for most purposes
- provides only one active mechanism per host or user authentication
- unable to provide finer grained access control

INN V2 introduces new Perl script to allow finer grained control over news article posting.

In conjunction with LDAP data, the `filter_nnrpd.pl` script is used to provide finer grained newsgroup access.

Special local newsgroup project required:

- unrestricted read access by local users
- restricted post access by limited authorized users, and no use of INN “moderator” mechanism

Problems encountered:

- basic INN access and posting mechanisms could not satisfy requirements
- where to store information about limited authorized users

Resolution implemented in conjunction with `filter_nnrpd.pl` script:

- special INN userids plus passwords are defined in `nnrp.access`, INN “user authentication” file
- protected LDAP attributes defined to contain special INN “authorized userids”
- protected LDAP attributes defined to contain hostnames of “authorized users”
- LDAP ACL’s used to protect requisite LDAP attributes
- LDAP is used to store information about special local newsgroup including “owner”

-
- when an article is to be posted to special newsgroup, script uses LDAP data to determine access privileges
 - special newsgroup reply postings only allowed from “authorized user” host or INN authenticated “authorized userid”
 - Web-based LDAP access allows “owner” of special newsgroup to dynamically update authorized user data independent of INN operation
 - LDAP data and Web-based access to it provide a “contact spot” where users can obtain more information about special newsgroup, and who to contact about it

Sample INN Project LDAP

dn: cn=<local newsgroup name>,cn=Network News Service, ...

objectclass: service

objectclass: simpleSecurityObject

objectclass: localextra

cn: <local newsgroup name>

multilinedescription: special local newsgroup

category: Network News

mail: <mailbox>@<local network news server>

owner: cn=<Service Group>,ou=People, ...

certifiedhost: <specific local network host>

certifiedauthor: <INN userid>

[P5] “File Relay”

- Project initially developed for use between user workstations and IBM mainframe
- can be extended to other “remote hosts” in place of mainframe
- uses Samba to provide access to Windows-based PC's
- uses NetaTalk to provide access to Macintosh-based PC's
- access by “Unix” workstations also possible, though not included in this project
- maintains two log files: one system, one user

-
- uses Expect and Cron to automate job submission process
 - main purposes for facility:
 - alleviate user interaction and connection with “foreign host”
 - provide native/local access to service through “network attached” folders
 - provide automated job submission process, relieving users from tasks to perform the same function
 - abstract “foreign host” processes for job submission and data retrieval, simplifying tasks for users
 - reduce time users spent in performing similar tasks on their own
 - provide possibility of E-mail notification for job submission problems

-
- LDAP used to store information about each user of service
 - LDAP used to store security information required for service to access “foreign host” on behalf of user
 - LDAP used to store e-mail address for possible job submission problem notifications
 - LDAP Access Control Lists used to protect security related attributes
 - users set their own “foreign host” security information through Web-based interface
 - “foreign host” job submission and output retrieval based upon FTP; others possible

-
- user creates “foreign host” job file on their workstation
 - user copies job file to server “job submission network folder”
 - automated server function “submits” user job file to “foreign host”
 - user job files that include FTP commands to server, copies file to server “output folder”
 - user copies data output from “network data output folder” to other local workstation folder

“File Relay” LDAP

dn: cn=<user name>, ou=<dept name>, ...

objectclass: umichPerson

objectclass: person

objectclass: organizationalPerson

objectclass: localExtra

objectclass: localNetwork

cn: <user name>

cn: <user.name>

sn: <lastname>

uid: <whatever>

userpassword: <LDAP password>

mail: <mailbox>@<mailhost>

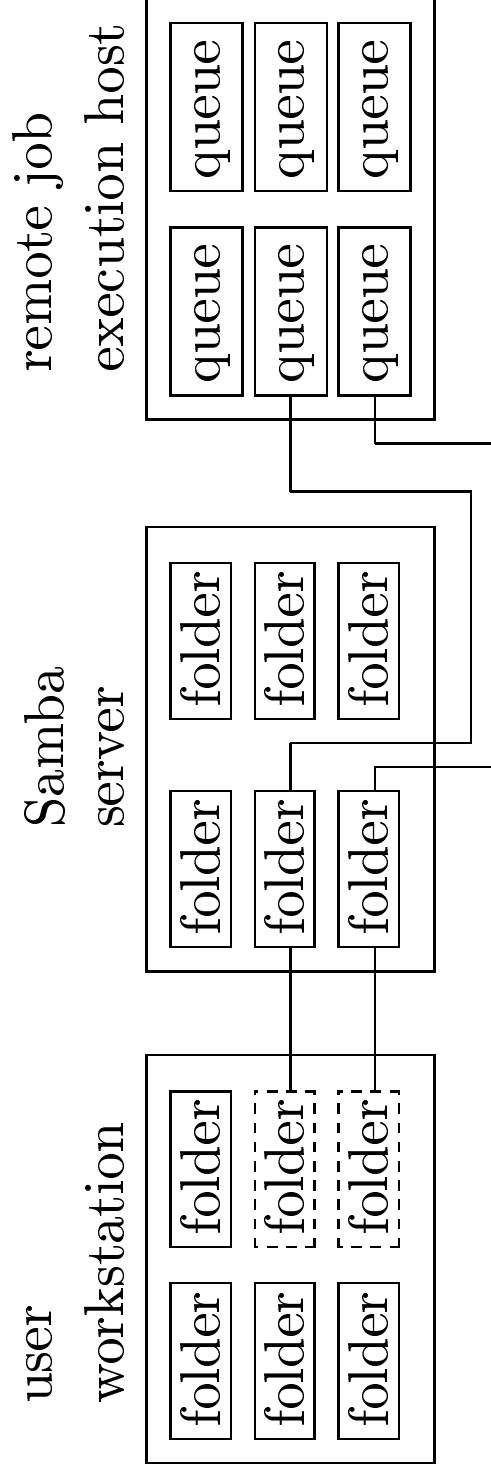
mvsuserid: <foreign host userid>

mvspasswd: <foreign host password>

mvsnotify: <notification mailbox>@<mailhost>

netbiosname: <non-IP workstation name>

“File Relay” End Results



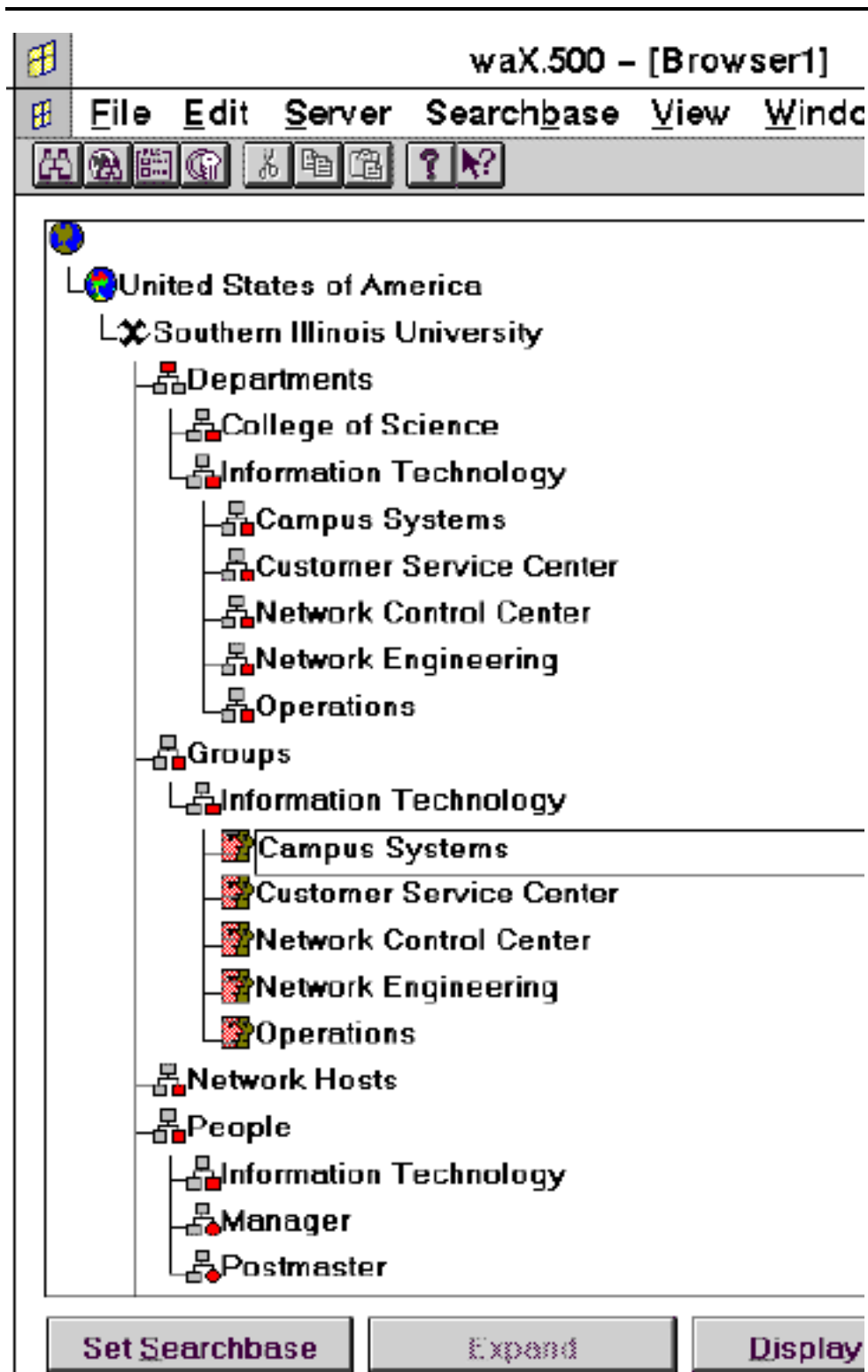
User to Samba server

Samba server does all of the work

LDAP Suggestions

- implement as many test LDAP servers as possible; have any spare 486/586/P5 PC's?
- restructure LDAP database(s) many times
- implement multiple production LDAP servers, even if not equal in size
- make sure that LDAP structure and use remains flexible and modifiable after put into production
- use new/local names for new objects/attributes; don't add to existing object/attribute definitions
- “normalize” the data; may require multiple iterations

-
- make use of multiple indexes, but pay attention to potential costs
 - seek out and develop relationships with different departments, even if only doing simple tests
 - obtain a copy of the *Programming Directory-Enabled Applications with LDAP* book; it is a very good reference
 - test the UMich WAX/MAX500 (Windows/Macintosh) clients - they have very nice hierarchical displays



Availability and Summary

Scripts and other goodies are at:

<http://www.usenix.org/events/usenix2000/freenix/dutton.html>

- P1. LDAPSEARCH and dynamic HTML LDAP query for LAN Admin info
- P2. JavaScript-based Web page with “raw” output format to display user host info
- P3. Sendmail service enhancements using LDAP
- P4. INN service enhancement using LDAP
- P5. “File Relay” using LDAP, cron, Expect, Perl, and Samba

[P6] 4-5,000 Mainframe Userid Search

(If time allows)