CU CS 4995-03: Internet Systems Programming

LDAP

Overview

- LDAP = networked access to databases and directories
- directory services
- LDAP overview
  - information model: information representation
  - naming model: organization and referencing
  - functional model: how to access and update
  - security model
- LDAP API

Network directory services

- white pages vs. yellow pages
- hierarchy vs. local
- structured (ASN.1, parameter/value, . . .) vs. full-text
- key-based retrieval (e.g., dbm) vs. SQL and kin
- applications:
  - email
  - web?
  - QoS policy: SLAs, transferred via COPS from PDP to PEP

Examples

protocol	application	problems
DNS	host-to-IP	restricted records, keymatch only
SLP	resources	LAN only
Corba COS	name to id	Corba only
LDAP	lots	performance
whois	domain contact	unstructured output
web search	search engine	unstructured output
WAIS	Z39.50	structured
Sun Jini		

LDAP background

- lightweight front-end to OSI X.500 directory access protocol; used by X.400 email
- runs over TCP (port 389)
- uses string formats for data elements
- uses ASN.1 encoding ($\boxed{\text{BER}}$, DER, PER encoding)
- LDAPv2: $\boxed{\text{RFC 1777}}$, RFC 1778; API in RFC 1823
- LDAPv3: RFC 2251, RFC 2252
- LDAP URLs: RFC 1959
- Corba IORs in LDAP: RFC 2714

LDAP implementations

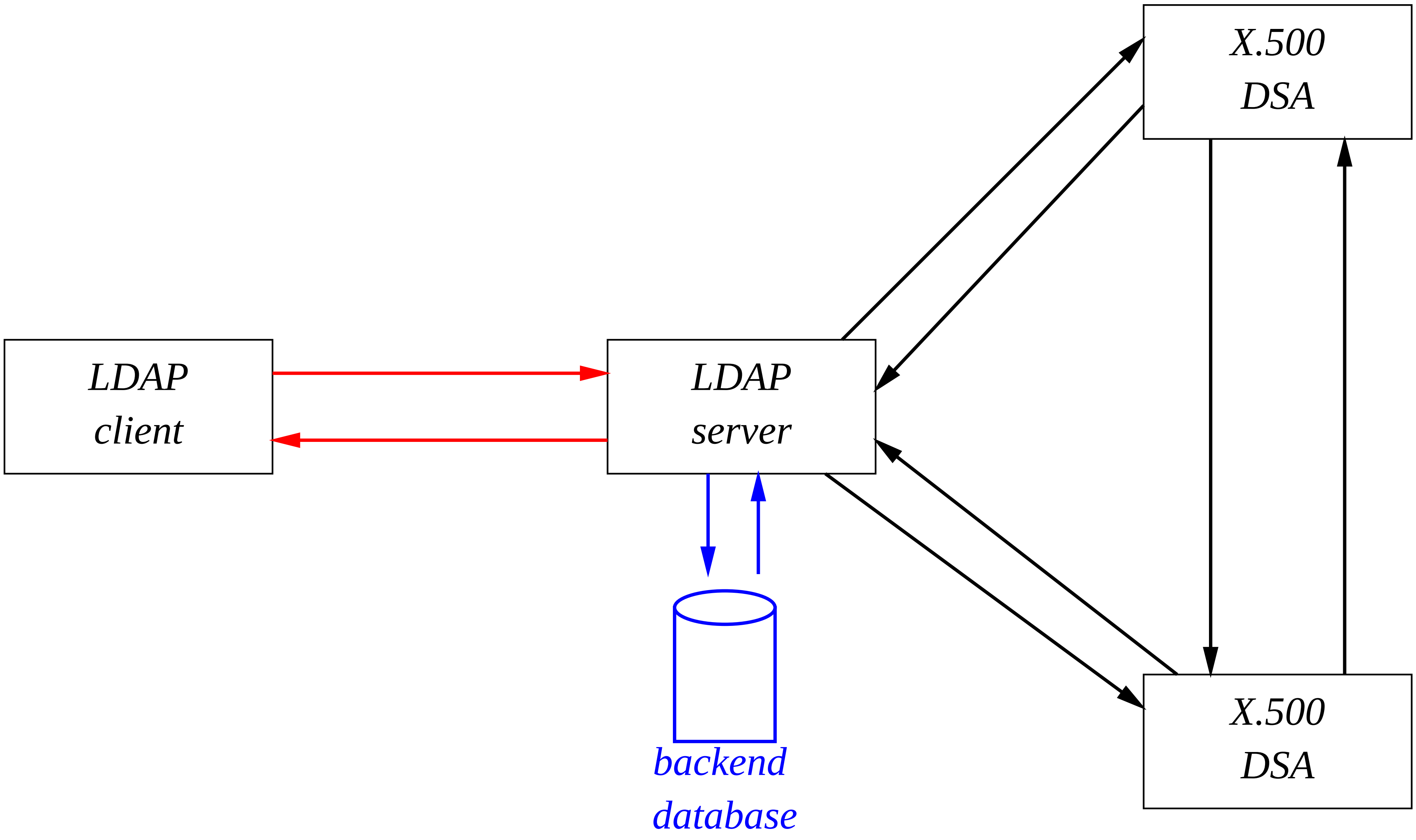
- U. Michigan `slapd` $\longrightarrow$ OpenLDAP
- Netscape, IBM, . . . servers
- Netscape client SDK for C and Java (free)

all need a *back end*, e.g., Berkeley db

Example: Microsoft Actice Directory

- DNS servers locate replicated LDAP servers
- LDAP stores global information
- including DCOM object identifiers (GUIDs)

LDAP architecture



Information model

- *entry* = information about object (person, printer, service class, ...)
- entry = $\geq$ 1 attributes
- attribute: type + value(s)
- attribute *syntax* restricts values and guides searches (e.g., caseIgnoreString, telephoneNumber)
- *content rules* limit size, attributes required, ...
- attribute ObjectClass: e.g., person $\approx$ database schema
- object classes can be added ("auxiliary") to entry
- LDAPv3 extensibleObject

Naming model

- generally, tree structure (many trees!)
- position identified by distinguished name (DN)
- *relative distinguished name* (RDN)
- leaf and non-leaf nodes may contain content
- “little endian”
- *alias* entries

Distinguished names

- RDN (attribute + value), separated by "," or ";"

```
o=Columbia University, c=US
o="Ace Industry, Inc.", c=US
cn=Babs Jensen + l=Cayman Islands, o=Privacy Bank,c=US
```

Functional model

Interrogation: search, compare with *search filter*;

- search entry, children or tree
- referrals
- specify size and time limits

Update: add, delete, modify, modify RDN (same server, tree position)

Authentication: bind (null, password, SSL, Kerberos, . . .)

Termination: unbind (abandon lengthy search)

Security model

- bind operation introduces querier
- LDAP does not mandate security mechanisms
- Netscape, UMich: entries, attributes by DN, IP address, domain name

LDAP: protocol

```
SearchRequest ::= [APPLICATION 3] SEQUENCE {
  baseObject      LDAPDN,
  scope           ENUMERATED {
    baseObject              (0),
    singleLevel             (1),
    wholeSubtree            (2) },
  derefAliases    ENUMERATED {
    neverDerefAliases       (0),
    derefInSearching        (1),
    derefFindingBaseObj     (2),
    derefAlways             (3) },
  sizeLimit       INTEGER (0 .. maxInt),
  timeLimit       INTEGER (0 .. maxInt),
  typesOnly       BOOLEAN,
  filter          Filter,
  attributes      AttributeDescriptionList }
```

LDAP: protocol

```
Filter ::= CHOICE {
 and             [0] SET OF Filter,
 or              [1] SET OF Filter,
 not             [2] Filter,
 equalityMatch   [3] AttributeValueAssertion,
 substrings      [4] SubstringFilter,
 greaterOrEqual  [5] AttributeValueAssertion,
 lessOrEqual     [6] AttributeValueAssertion,
 present         [7] AttributeDescription,
 approxMatch     [8] AttributeValueAssertion,
 extensibleMatch [9] MatchingRuleAssertion }
```

Referrals

- *chaining*: ask server (cf. DNS)
- *referral*: contact another server
- LDAPv2: in error message
- LDAPv3: sequence of LDAP URLs

LDAP API

ldap_

search()	search for entries
compare()	does entry contain attribute?
bind()	authenticate
unbind()	terminate session
modify()	make changes to entry
add()	add entry
delete()	delete entry
modrdn()	modify RDN
result()	return result

Typical LDAP program

1. initialize library, obtain session handle: ldap_init() $\longrightarrow$ opens connection
2. initiate operation, wait for results
3. process results: ldap_get_values; ldap_first_entry, ldap_next_entry iterators with ldap_get_dn; sort with ldap_sort_entries
4. dispose of session handle: ldap_unbind()

Asynchronous LDAP

- protocol is asynchronous (multiple pending requests)
- ldap_search_s(): synchronous equivalent

LDAP search filters

- search filters are strings at library level, ASN.1 at protocol level
- equality, approximate (sounds like), substring, greater/less than, presence
- combine with AND, OR, NOT

```
(cn=*mary*)   // mary somewhere
(cn>=Fred)    // lexicographic
(&(objectclass=person)(mail=*))   // with email
(&(objectclass=person)
  (|(title=*professor*) (title=*dean*)))
```

LDAP URLs

`ldap://hostport/dn?attributes?scope?filter`

Examples:

- `ldap://ldap.columbia.edu/cn=Peter%20Jensen,o=...,c=US`
- `ldap://.../o=Columbia%20University?drink?sub?sn=Jensen`

LDAP tools

- ldapsearch
- ldapmodify
- ldapadd
- ldapdelete
- ldapmodrdn

useful for cgi scripts!

LDIF: LDAP Interchange Format

- textual format for interchange
- used by ldapsearch and ilk, Netscape address book
- continuation lines
- base-64 for binary

```
dn: cn=Robert Jensen, o=Columbia University, c=US
cn: Robert Jensen
cn: Bob Jensen
sn: Jensen
drink: scotch
mail: rj348@columbia.edu
objectClass: top
objectClass: person
```

LDAPv2 $\longrightarrow$ LDAPv3

- attribute inheritance (match across)
- operational attributes (modifiersName, modifiedTime)
- schema discovery
- security (SASL, used by IMAP, POP, NNTP)
- I18N
- referrals

Directory issues

- dependencies?
- performance?
- active directories: make up answers on the fly