(Open)LDAP aus Client-Sicht

Protokoll und OpenLDAP client library

RBG-Seminar SoSe 2007
12.06.2007

Holger Kälberer

Agenda

- Überblick: LDAP-Konzepte
- Client-Perspektive: Protokoll & libldap
- Beispiele: Einige Client-Anwendungen

Überblick: LDAP-Konzepte

LDAP

- ***L*ightweight** – “Erleichterung” von X.500 DAP; Uni. Michigan 1993; TCP/IP-basiert
- ***D*irectory** – optimiert fuer Lesezugriffe; (vgl. DB); hierarchisch
- ***A*ccess** – Authentifizierung; komplexe Filter; Authorisierung durch komplexe ACLs
- ***P*rotokoll** – v1 1993; v2 (RFC 3494) 1995; v3 (RFC 3377) 1997

DIT, Eintraege

- hierarchischer Verzeichnisbaum (mit frei waehlbarem namingContext)
- eindeutige Einträge durch DN (RFC4514); RDN
- Verzeichniseinträge ...
  - enthalten typisierte Attribute
  - ein (beliebiges) name-value-Paar = RDN
  - besondere: aliases, referrals, cn=config, subentries, ...
  - instanziieren Objektklassen

schemas

- Schemata definieren ...
  - Objektklassen (hierarchisch; structural|mixin|abstract; Member-Attribute)
  - Attributtypen
  - matchingRules
  - ldapSyntaxes (= 'einfache Datentypen', z.B. Directory String)
- OIDs
- vordefnierte in RFC 4519, u.a.; viele weitere verfuegbar

```
attributetype ( 1.3.6.1.4.1.26765.2.1.1001
   NAME 'mclSpamHandling'
   DESC 'How is spam handled for this
      email-recipient: {greylisting|rblisting}'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX  1.3.6.1.4.1.1466.115.121.1.15 )
```

```
objectclass ( 1.3.6.1.4.1.26765.2.2.1000
   NAME  'mclEmailSettings'
   SUP top
   AUXILIARY
   DESC  'Additional mcl-related information
      for email-handling.'
   MAY   ( mclSpamHandling )   )
```

Operationen, Erweiterungen

- Operationen client -> Server
  - un/bind, add, delete, search, modify, modifyDn, compare, abandon, startTLS
- RFC 4521
- extension: generische Protokoll-Operation (ExtendedRequest)
- controls, features (?): Mechanismus zur Erweiterung von LDAP-Operationen; z.B.:

```
1.3.6.1.4.1.4203.1.9.1.1: Sync Request Control (LDAP Content
Synchronization Control, RFC 4533)
2.16.840.1.113730.3.4.2: ManageDSAiT Control (RFC 3296)
1.3.6.1.1.14: Modify-Increment Extension (“feature”, RFC4525)
```

DSA

- DIT-Partitionierung -> (subordinate, superior) Referrals
- Replikation: Master -> Slave/s
- Security:
  - bind (ID, anonym) -> auth (Klartext, SASL)
  - TLS
  - ACLs (RFC 2820; 2000; informational)
- abfragbare DSA-Attribute

```
altServer, namingContexts, supportedControl, supportedExtension,
supportedFeatures, supportedLDAPVersion, supportedSASLMechanisms,
(maybe: subschemaSubentry, configContext)
```

OpenLDAP DSA

- LDAPv3 seit 2.0 (2000)
- statische oder dynamische config (seit 2.3)
- modular: frontend <-> (multi-)backends
- flexible ACLs (auch dynamisch)
- div. overlays
- Replikation:
  - bisher: openldap slurpd (gepflegt bis 2.1)
  - syncrepl (RFC 4533; 2006; exp.; ctrl-Erweiterung des SearchRequest)
- LDAP proxy cache
- C-, C++, TCL-API
- tools

Spezifikationen

RFCs 4510-4533

Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map [**RFC4510,** Jun 2006, Zeilenga, prop. standard]

- LDAP: The Protocol [**RFC4511**] (RFC2251; Dec. 1997)
- LDAP: Directory Information Models [**RFC4512**]
- LDAP: Authentication Methods and Security Mechanisms [**RFC4513**]
- LDAP: String Representation of Distinguished Names [**RFC4514**]
- LDAP: String Representation of Search Filters [**RFC4515**]
- LDAP: Uniform Resource Locator [**RFC4516**]
- LDAP: Syntaxes and Matching Rules [**RFC4517**]
- LDAP: Internationalized String Preparation [**RFC4518**]
- LDAP: Schema for User Applications [**RFC4519**]

obsoletes RFC 3377 (Sep 2002), 2251-2256, 2829, 2830, 3771)

Ueberblick: Anwendungen

- System-Integration: PAM/NSS, automount, lprng, samba
- WWW: apache-auth, apache-config
- DNS: Zoneninformationen; bind, pdns
- DHCP: komplette config
- squid
- Mail: aliases, *transport-maps, *-maps, s.u.
- Verwaltung: UCS verwaltet ganze Netzwerke ueber ldap; config/install-rollouts

Client-Perspektive: Protokoll & OpenLDAPs libldap

Grundsätzliches

- atomare Operationen
- asynchrones Verhalten
- einheitlicher envelope: LDAPMessage (msgID, protocolOp, controls)
- Ergebnis: LDAPResult (resultCode, matchedDN, diagnosticMsg, referral) = return der libldap-calls
- Server muss fuer Konformitaet mit directory model und verfuegbaren Schemata sorgen

ldap.h: Uebersicht

Initialisierung

```
LDAP *ldap_open(char *host, int port);
LDAP *ldap_init(host, port);
ldap_initialize(LDAP **ld, char *url);

ldap_set_option(LDAP *ld, int option, LDAP_CONST void
                *invalue)
```

Operationen

s.u.

Ergebnis-verarbeitung

```
LDAPMessage *ldap_first_message(LDAP *ld, LDAPMessage
                                *result);
LDAPMessage *ldap_next_message(LDAP *ld, LDAPMessage
                               *message);
int ldap_count_messages(LDAP *ld, LDAPMessage
                        *result);
```

ldap.h: Uebersicht (Forts.)

```
char *ldap_first_attribute(LDAP *ld, LDAPMessage
                        *entry, BerElement **berptr);
char *ldap_next_attribute(LDAP *ld, LDAPMessage
                           *entry, BerElement *ber);
char **ldap_get_values(LDAP *l, LDAPMessage *entry,
                         char *attr);
struct berval **ldap_get_values_len(LDAP *ld,
     LDAPMessage *entry, char *attr);
```

BER Library (lber.h)

lber-decode(3)
lber-encode(3)

Tools

ldap_sort(3)
ldap_url(3)

ldap.h: Funktionsvarianten

- synchron vs. asynchron + einfach vs. extended (controls); z.B.:

```
int ldap_add(LDAP *ld, const char *dn, LDAPMod *attrs[]);
int ldap_add_s(LDAP *ld, const char *dn, LDAPMod *attrs[]);
int ldap_add_ext(LDAP *ld, const char *dn, LDAPMod *attrs[],
    LDAPControl *sctrls[], LDAPControl *cctrls[], int *msgidp);
int ldap_add_ext_s(LDAP *ld, const char *dn, LDAPMod *attrs[],
     LDAPControl *sctrls[], LDAPControl *cctrls[]);
```

- Verarbeitung asynchroner calls:

```
int ldap_result(LDAP *ld, int msgid, int all, struct timeval
                *timeout, LDAPMessage **result );
int ldap_msgtype( LDAPMessage *msg );
```

Operationen

RFC 4511	*ldap.h*
-> BindRequest (Version, BindDN, authMethode) (RFC4513) <- BindResponse (LDAPResult)	int ldap_bind(LDAP *ld, const char *who, const char *cred, int method);
-> UnbindRequest()	int ldap_unbind(LDAP *ld);
-> SearchRequest (BaseDN, scope, derefAliases, sizeLimit, timeLimit, typesOnly, filter, attributes) <- SearchResult(SearchResultEntries, SearchResultReferences, SearchResultDone)	int ldap_search(LDAP *ld, char *base, int scope, char *filter, char *attrs[], int attrsonly);

Operationen (Forts.)

```
-> ModifyRequest(ModDN, seq of add|
delete|replace-operations,
modification)
<- ModifyResponse (= LDAPResult)
```

```
int ldap_modify(LDAP *ld, char *dn,
LDAPMod *mods[];
```

```
-> AddRequest (AddDN, attributes)
<- AddResponse (=LDAPResult)
```

```
int ldap_add(LDAP *ld, const char *dn,
LDAPMod *attrs[]);
```

```
-> DelRequest (DN)
<- DelResponse = LDAPResult
```

```
int ldap_delete(LDAP *ld, char *dn);
```

```
-> ModifyDNRequest (ModDN, newRDN,
delOldRDN, newSuperior)
<- ModifyDNResponse = LDAPResult
```

```
int ldap_modrdn2(LDAP *ld, char dn,
char *newrdn, int deleteoldrdn);
```

Operationen (Forts.)

```
-> CompareRequest (compDN,
attrValAssertion)
<- CompareResponse (= LDAPResult)
```

```
int ldap_compare(LDAP *ld, char *dn,
char *attr, char *value);
```

```
-> AbandonRequest (MsgID)
```

```
int ldap_abandon(LDAP *ld, int msgid);
```

```
-> ExtendedRequest (reqOID,reqValue)
<- ExtendedResponse (seq of
(LDAPResult, respOID, respValue))
```

```
-> StartTLS=ExtendedRequest(1.3.6.1.
4.1.1466.20037, NULL)
<- StartTLSResponse=ExtendedResponse
(LDAPResult, 1.3.6.1.4.1.1466.20037,
NULL)
```

```
int ldap_start_tls_s(LDAP *ld,
LDAPControl **serverctrls, LDAPControl
**clientctrls );
```

```
<- IntermediateResponse (respName,
respValue)
```

SearchRequest

- Parameter: BaseDN, scope, derefAliases, sizeLimit, timeLimit, typesOnly, filter, attributes
- Filter (RFC 4515)
  - Typen: equalityMatch, substrings, ge, le, present, approxMatch, extMatch

```
"uid=*"
"objectclass=posixGroup"
"(&(objectclass=posixGroup)(memberUid=hkaelber))"
"(&(objectClass=mclEmailSettings)(|(mail=%u@math.uni-bielefeld.de)
                                   (mail=%u@mathematik.uni-bielefeld.de)))"
```

- root DSE Attribute erfragen:

```
$ ldapsearch -x -s base -b "" "(objectClass=*)" -h vldap1 -LLL namingContexts
dn:
namingContexts: dc=math,dc=uni-bielefeld,dc=de
```

SearchRequest (Forts.)

```
LDAP *session;
LDAPMessage **ldap_results; LDAPMessage *ldap_msg;
char *ldap_attr;
char **ldap_vals;
char *attrs[3] = {"cn", “uidNumber”, NULL};
if ((session = ldap_open(“localhost”, 389)) != NULL)
  if (ldap_simple_bind_s(session, NULL, NULL) ==
      LDAP_SUCCESS)
    if (ldap_search_s(session “ou=people,dc=india,dc=de”, LDAP_SCOPE_SUBTREE,
        “uid=*”, attrs, FALSE, ldap_results) == LDAP_SUCCESS)
      for (ldap_msg = ldap_first_message(session, ldap_results); ldap_msg != NULL;
           ldap_msg = ldap_next_message(session, ldap_msg))
        for (ldap_attr = ldap_first_attribute(session, akt_msg, &berptr);
         ldap_attr != NULL; ldap_attr = ldap_next_attribute(session, akt_msg, berptr))
          if ((ldap_vals = ldap_get_values(session, akt_msg, ldap_attr)) != NULL)
            for (i=0; i<ldap_count_values(ldap_vals); i++)
            {
              printf("%s: %s\n",ldap_attr, *(ldap_vals+i));
              attr_num++;
            }
//...
```

SearchRequest (Ergebnis)

```
cn: Holger Kaelberer
uidNumber: 1000
cn: Gast1
uidNumber: 1001
cn: Hanna Hoenicke
uidNumber: 1002
cn: Sonia Filippini
uidNumber: 1003
cn: Konrad Waidmann
uidNumber: 1004
```

Anforderungen an Clients

- Referrals & continuations
  - follow (superior schreibend, subordinate lesend)
  - Schleifen vermeiden
- ggf. asynchron rufen
- ggf. controls abfragen und auswerten
- ggf. um TLS/SASL kuemmern
- um besondere entries kuemmern (userPassword, Fotos, usw.)

Einige Client-Anwendungen

1. Postfix (1): Mail-Aliase

```
alias_maps =    ldap:/usr/local/etc/postfix/ldap-aliases.cf
```

```
server_host = XYZ.math.uni-bielefeld.de ABC.math.uni-bielefeld.de
search_base = ou=aliases,dc=math,dc=uni-bielefeld,dc=de
query_filter = (&(objectClass=nisMailAlias)(cn=%s))
result_attribute = rfc822MailMember
```

```
# postmap -q testlist ldap:/etc/postfix/ldap-aliases.cf
hkaelber
juser
```

2. Postfix (2): Nutzerbasiertes Spamhandling

```
smtpd_restriction_classes = greylisting
greylisting = check_policy_service inet:127.0.0.1:2501
...
smtpd_recipient_restrictions =
...
   check_recipient_access ldap:/usr/local/etc/postfix/ldap-spamhandling.cf,
...
```

```
server_host =  XYZ.math.uni-bielefeld.de
search_base = dc=math,dc=uni-bielefeld,dc=de
bind_dn = XYZ
bind_pw = XYZ
query_filter = (&(objectClass=mclEmailSettings)(|
(mail=%u@math.uni-bielefeld.de)(mail=%u@mathematik.uni-
bielefeld.de)(cn=%u)))
result_attribute = mclSpamHandling
```

```
...
objectClass: mclEmailSettings
mclSpamHandling: greylisting
...
```

3. Postfix (3): Kaskaden

```
cn=agroup, dc=example, dc=com
objectclass: top
objectclass: ldapgroup
cn: agroup
mail: agroup@example.com
memberdn: uid=auser, dc=example, dc=com
memberdn: uid=buser, dc=example, dc=com
memberaddr: auser@example.org
memberaddr: buser@example.org
```

```
dn: uid=auser, dc=example, dc=com
objectclass: top
objectclass: ldapuser
uid: auser
mail: auser@example.com
maildrop: auser@mailhub.example.com

dn: uid=buser, dc=example, dc=com
objectclass: top
objectclass: ldapuser
uid: buser
mail: buser@example.com
maildrop: buser@mailhub.example.com
```

```
special.cf:
        ...
    search_base = dc=example, dc=com
    query_filter = mail=%s
    result_attribute = maildrop
    special_result_attribute = memberdn
$ postmap -q agroup@example.com ldap:special.cf
auser@mailhub.example.com,buser@mailhub.example.comc
```

Danke!