# LDAP

## Adattárolás OpenLDAP programmal

2006. 05. 03

*Jónás Zsolt*

## Változások listája:

- ✗ 2006. 05. 03
  - ✔ első változat

# Tartalomjegyzék

## Előszó

A vállalatok életében jelentős szerepet tölt be az informatika. A felhasználói szám növekedésével egyre több adatot kell „kézben tartani”, szinkronizálni. A cégösszevonások és átcsoportosítások szintén növelik a rendszer összetettségét. Az adminisztráció segítéséhez és a konzisztencia megőrzéséhez megalkották a címtárakat. Ezek segítségével egy helyre csoportosítva végezhetjük el a kívánt változtatásokat, valamint a felhasználók élete is könnyebbé válik az egyszeri azonosítás (*Single Sign On*) alkalmazásával.

Más-más platformon több cég is kínál megoldást a problémára. Mi egy GPL licence alatt elérhető programmal fogunk megismerkedni Debian GNU/Linux operációs rendszeren. Természetesen a program megtalálható más linux disztribúcióban is, de a különböző terjesztések specialitásából adódó különbségekre külön-külön nem térünk ki, bár ezek inkább csak az adott fájlok elérési útvonalában realizálódnak elsősorban, azonos verziószám esetén a működés megegyezik.

# 1. Bevezető

TODO

## 1.1. Használt jelölések

A könnyebb érhetőség kedvéért különböző betűtípusokkal szedünk bizonyos szavakat, szövegrészleteket, szövegblokkokat.

```
Példa
```

`Parancsok`

*Kiemelés*

Sortörés: „\”

[meg lehet adni]

<meg kell adni>

# 2. Alapok

Az évek folyamán egyre nagyobb lett az információs éhség. Évről-évre, napról-napra egyre több információt kell tárolni. A különböző adattárolások két fő típusba sorolhatók:

- relációs adatbázis,
- címtár.

A két fő típus alapvető dolgokban térnek el egymástól, így nem egymás kiváltására alkották meg őket, hanem egymás kiegészítése érdekében.

A relációs adatbázisok mellett a címtárakat is több alkategóriába sorolhatjuk, melyek a következők:

- operációs rendszer része (felhasználók és adataik, csoportok),
- alkalmazás specifikus (csak egy adott alkalmazás használja, elszeparálva),
- cél specifikus (egy kimondott feladat elvégzésére szolgál, például DNS),
- általános (egyszerre több alkalmazás igényeit is képes kielégíteni hatékonyan, például az LDAP).

Minél több szolgáltatást és programot használnak egy vállalaton belül, minél több helyen használunk felhasználó-azonosítást, annál nehezebb rendszergazdaként kézben tartani a felhasználók azonosítását. Nagy segítséget jelent egy központosított azonosítási rendszer. Az LDAP rugalmassága és széleskörű felhasználhatósága képessé teszi, hogy alkalmas legyen a központi menedzsment által támasztott követelményeknek megfelelni.

## 2.1. Címtárakról

Kezdetben a címtárakban a felhasználókat és azok adatait tárolták (felhasználónév, jelszó, telefonszám, e-mail cím stb.). Manapság több mindent, többféle adatot lehet tárolni bennük, de összességében mindegyikre igaz, hogy az olvasás gyakoriságához képest ritkán változnak. Ezen oknál fogva a címtárak elsősorban keresésre és listázásra optimalizálta, így az írási műveletek viszonylag több időt vesz igénybe, valamint az adatokat hierarchikus formában tárolják.

Mivel nem a folytonos írás/módosítás a fő szempont, ezért a címtárak általában nem alkalmaznak bonyolult *tranzakció kezelést* vagy *roll-back* rendszereket, ezek a funkciók inkább az adatbázis- kezelők sajátosságaik. A cél, hogy az összetett kérdésekre is gyorsan válaszoljanak. A minél nagyobb rendelkezésre állás érdekében, lehetőség van replikák (többszörözések) üzembe állítására.

Azonban hiába a gondos tervezés és megvalósítás, ettől még nem lesz egyik címtár se csodafegyverek. Bizonyos bonyolultság és méret felett, eleve nem lehet vagy érdemes csak egy címtárat használni. Saját rendszerünkben is lehetnek olyan programok, alkalmazások, amelyek más-más címtárat követelnek meg, nem beszélve esetleg a különböző osztályok összekapcsolásáról, cégfelvásárlásokról, cégösszevonásokról. A kialakított rendszereket nem biztos, hogy könnyen össze lehet olvasztani, de előfordulhatnak ideiglenes partnerkapcsolatok is, melyek esetén nem célszerű a két cég címtárát egy címtárrá egyesíteni. Sok egyéb indok is előfordulhat, ami miatt nem akarjuk vagy tudjuk egyesíteni az adatokat egy címtárba. E probléma orvoslására alkották meg a *virtuális címtárakat* és a *metacímtárakat*.

### 2.1.1. Virtuális címtárak

Ahogy a neve is utal a működésére, használatával nem hozunk létre újabb címtárat, csak elfedjük őket a felhasználók elől. Egy egységes felületet hozunk létre a többi, különböző címtár fölé. Ha kérés érkezik hozzá – a beállítások alapján – továbbítja a megfelelő címtárnak. A probléma abban áll, hogy a bejövő kéréseket át kell fordítani és optimalizálni a cél címtár „nyelvére”, valamint hogy hogyan tartsuk fenn a konzisztenciát. Ennek érdekében célszerű bevezetni bizonyos szűréseket, megkötéseket, hogy csak a mindenhol támogatott elemeket használhassuk.

### 2.1.2. Metacímtárak

A virtuális címtárakkal ellentétben, nem csak egy felület a többi címtár felett. A csatolt adatforrások fontosabb adataiból létrehoz egy új címtárat. Figyeli a csatolt címtárakban a változásokat, és annak megfelelően intézkedik, hogy megtartsa a konzisztenciát. A nehézséget az jelentheti, hogy meg tudjuk-e találni azokat az objektumokat a különböző adatforrásokban, amelyek ugyanazt az elemet (felhasználó, csoport stb.) jelentik.

A metacímtárak egyik tipikus felhasználási területe, amikor több cég szeretne közösen dolgozni, de a külsős embereket nem akarják felvenni a belső vállalati címtárba.

## 2.2. Mi is az az LDAP?

Az LDAP a „Lightweight Directory Access Protocol” (egyszerű címtár hozzáférési protokoll) kifejezést takarja. Ahogy a neve is sugallja, egy kliens-szerver protokollról van szó, amelyet címtárak elérésekor használnak.

Kezdetben több különböző címtár is létezett, de nem volt egy egységes felület, ezért több ajánlás egységesítésével létrehozták az X.500 (DAP – Directory Access Protocol) protokollt, azonban számos hátránya volt:

- A teljes OSI protokollal dolgozik, a TCP/IP helyett.
- Akkoriban az OSI protokoll készlet (stack) használata nagy erőforrást igényelt.
- Feleslegesen túlbonyolított protokoll lett, ezért nehéz volt megvalósítani.

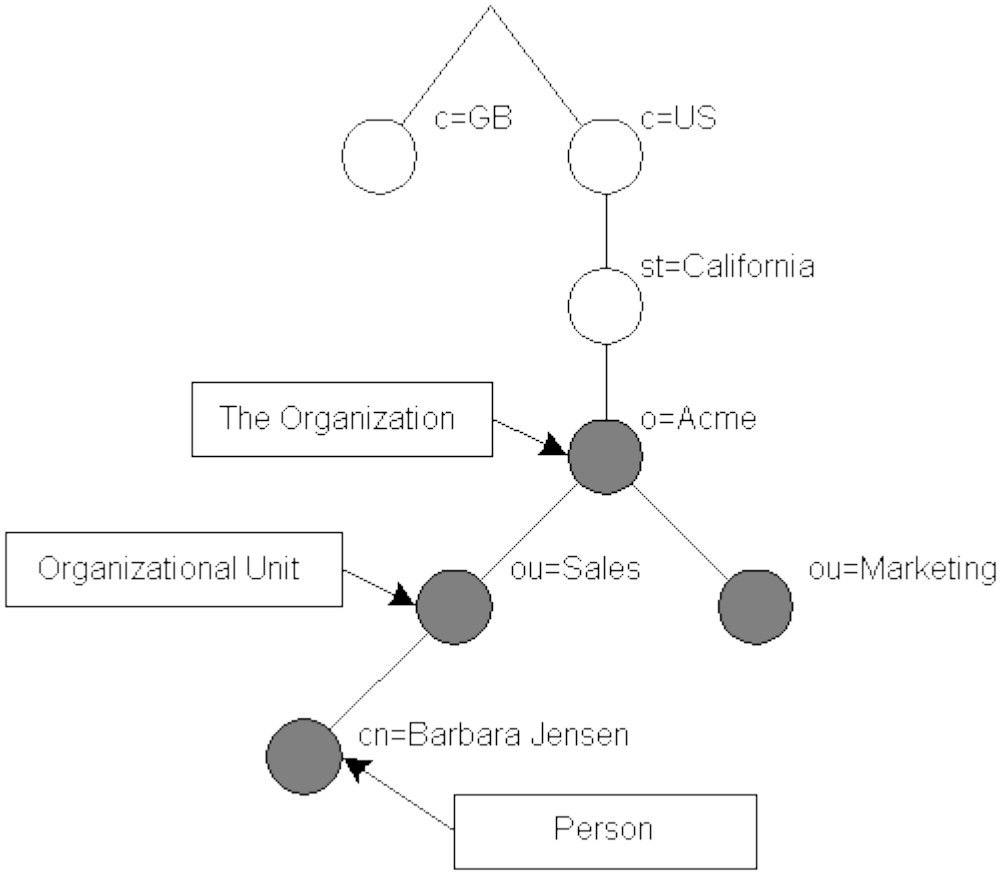
A protokoll hátrányai miatt a kliensek nem csatlakoztak közvetlenül a DAP kiszolgálókhoz, hanem csak egy adott átjáróhoz. A kliens gépek egyszerűsített protokollokon keresztül kommunikáltak a különböző átjárókkal, amelyek átfordították és továbbították a kéréseket és a DAP kiszolgálóknak.

Az információk fastruktúra (hearhikus) szerkezetben tárolódnak, melyben minden csomópont egy *bejegyzést* (*entry*) jelent. A bejegyzés által „*kötelezően tárolandó*” (*must*) vagy „*tárolható*” (*may*) információit a bejegyzést *típusai* (*objectclass*) határozzák meg. A bejegyzésekre a saját „*egyedi nevével*” (*DN, Distinguished Name*) hivatkozhatunk, amely megegyezik a tőle a fa csúcsához (*top*) vezető úttal.

A hagyományoknak megfelelően *(2.2.1. ábra: LDAP-fa (hagyományos elnevezés))*, a hierarchikus elrendeződést a területi és/vagy a vállalati tagolódás határozza meg. Az alábbi példa a következőt szemlélteti:

- a struktúra tetején szerepelnek az országok ('*c*'),
- adott ország tartalmaz államokat, megyéket stb. ('*st*'),
- azon belül vannak vállalatok ('*o*'),
- vállalaton belül lehetnek különböző szervezeti (users, groups, stb.) egységek ('*ou*'),
- szervezeti egységeben belül pedig elemek különböző tulajdonságokkal ('*cn*', '*uid*', stb.).

Szemléltetésként álljon itt az OpenLDAP dokumentációjában szereplő ábra:




*2.2.1. ábra: LDAP-fa (hagyományos elnevezés)*

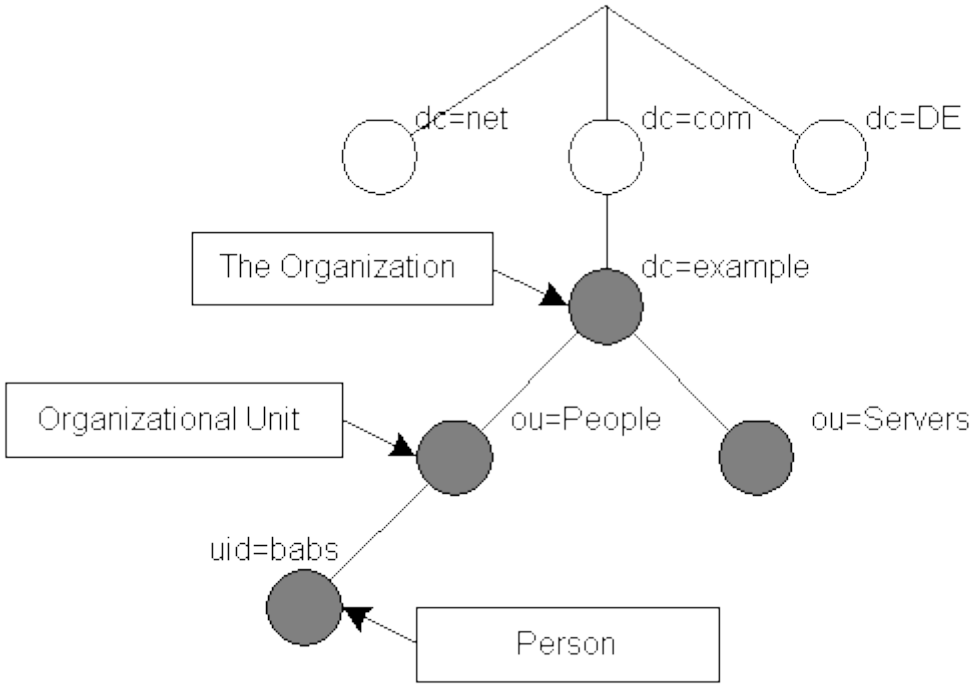
Ebben a hagyományos elnevezési struktúrában a „*cn=Barbara Jensen*” bejegyzés egyedi neve (*DN*) a következő: „*cn=Barbara Jensen,ou=Sales,o=Acme,st=California,c=US*”.

Létezik egy másik elnevezési rendszer, amely manapság egyre népszerűbb, ez az Internetes domain nevek megadása, melyet a *2.2.2. ábra: LDAP-fa (internetes elnevezés)* szemléltet. Felépítése a következő:

- a struktúra tetején szerepel a vállalat internetes neve, pontonként elválasztva ('*dc*'),
- vállalaton belül lehetnek különböző szervezeti (users, groups, stb.) egységek ('*ou*'),
- szervezeti egységeben belül pedig elemek különböző tulajdonságokkal ('*cn*', '*uid*', stb.).

Ennél az elnevezésnél szokás a fa csúcsára (*top*) tenni a vállalat teljes domain nevét: „*dc=proba,dc=hu*” (*dc, domain component*) és egyben ide szokták megadni a vállalat nevét is ('*o*'). Természetesen ez mind függ a vállalat méretétől, földrajzi tagolódásától stb.

Szemléltetésként álljon itt az OpenLDAP dokumentációjában szereplő ábra:




*2.2.2. ábra: LDAP-fa (internetes elnevezés)*

Ebben a hagyományos elnevezési struktúrában az „*uid=babs*" bejegyzés egyedi neve (*DN*) a következő: „*uid=babs,ou=People,dc=example,dc=com*".

Az egyedi név (*DN*) két részből tevődik össze:

- a bejegyzés saját, viszonylagos, *relatív nevéből* (*RDN, Relative Distinguished Name*),
- valamint a szülő bejegyzés egyedi nevéből (*DN*).

Az előbbi példánál maradva:

- RDN: uid=babs,
- DN: uid=babs,ou=People,dc=example,dc=com

### 2.2.1. Főbb különbségek az LDAPv2 és LDAPv3 között

Az LDAP-átjárók népszerűsége idővel egyre jobban nőt, minek következtében megjelent az első natív LDAP címtár, amely már nem igényelt DAP kiszolgálót. A fejlődés nem állt meg és két évre rá megjelent a következő generáció (v3), amely manapság egyet jelent „a címtár” fogalmával.

A hármas verziót a '90-es évek végén fejlesztették ki, hogy leváltsa a nagy elődöt, a kettes szériát. Fontos újdonságokat hozott, többek között:

- SASL és TSL támogatása,
- az adatokat UTF-8 kódolással tárolja,
- átirányíthatók a lekérdezések,
- lekérdezhetővé vált a használtban levő sémák listája,
- kiterjeszthetők a különböző parancsok, utasítások stb.

Kurrens változások történtek, ennek következtében nagyon hamar elterjedt, manapság szinte csak ezzel a változattal találkozni, mondhatni, teljesen kiszorította a piacról az előző változatot.

## 2.3. OpenLDAP

Az OpenLDAP (http://www.openldap.org) az LDAP egy nyílt forráskódú implementációja. Számos platformot támogat, így széles körben használható és lehetőség van különböző platformokon futtatni a kiszolgálókat és replikákat. A 2.0-ás változata óta támogatja az LDAPv3 ajánlást.

### 2.3.1. slapd

A `slapd` valósítja meg az LDAP címtár szolgáltatást, amelyben bármit tárolhatunk[1], amit csak akarunk. Sok hasznos képességgel bír, hogy igényeinknek minél jobban megfeleljen, többek között a következőkkel:

- LDAPv3 óta képes kommunikálni IPv4, IPv6 és Unix IPC csomagokkal.
- Támogatja az SASL azonosítást több mechanizmuson (DIGEST-MD5, EXTERNAL,

1 Bármit tárolhatunk benne, azonban ez nem jelenti az, hogy ajánlott is.

GSSAPI) keresztül a Cyrus SASL segítségével.

- ✗ TLS (és SSL) támogatása az OpenSSL programon keresztül.
- ✗ Unicode kódlapot használ az adatok tárolására.
- ✗ A hozzáférések korlátozása (access) sok lehetőséget ad kezünkbe. Korlátozhatók az azonosítási, listázási, olvasási, módosítási, és bővítési jogok akár külön-külön is az adatbázisban tárol bármelyik elem bármelyik tulajdonságára.
- ✗ Többféle adatbázis backend támogatása, többek között:
  - PASSWD, /etc/passwd fájl használata csak-olvasható módban;
  - LDAP, Lightweight Directory Access Protocol (Proxy);
  - SHELL, tetszőleges shell script;
  - PERL, tetszőleges perl script;
  - SQL, adatok tárolása SQL adatbázisban;
  - LDBM, Lightweight DBM;
  - BDB, Berkeley DB;
  - stb.
- ✗ Többszörös adatbázis támogatás: egy `slapd` szerver képes kiszolgálni egyszerre több, logikailag akár különböző LDAP-fához érkező kéréseket is, és ezen adatbázisokhoz akár különböző backend-ek tartozhatnak.
- ✗ Saját modulok illesztéséhez API.
- ✗ Képes üzemelni proxy-cache üzemmódban is.

### 2.3.2. slurpd

A `slurpd` egy daemon, amely a `slapd` programmal együtt nyújt replika szolgáltatást. A mester adatbázisban történt változásokat megbízható módon viszi át a különböző replikákhoz. Számon tartja a frissítéskor nem elérhető replikákat, és gondoskodik a változások későbbi érvényesítéséről, tehermentesítve ezáltal a `slapd` programot, hogy a replikákkal is foglalkoznia kelljen.

## 2.4. Hol található több információ az LDAP-ról?

Az LDAP alapját képező DAP ajánlásról az alábbi helyen olvashatunk:

- X.500 recommendations are available from the ITU (http://www.itu.int/rec/T-REC-X.500/)

Az LDAP 3-as verziója (LDAPv3) csak egy ajánlás, azonban több RFC is foglalkozik vele (http://www.ietf.org/rfc.html):

- RFC 3377: Lightweight Directory Access Protocol (v3): Technical Specification
- RFC 2251: Lightweight Directory Access Protocol (v3)
- RFC 2252: LDAPv3: Attribute Syntax Definitions
- RFC 1960: LDAP String Representation of Search Filters
- RFC 2253: LDAPv3: UTF-8 String Representation of Distinguished Names
- RFC 2254: The String Representation of LDAP Search Filters
- RFC 2255: The LDAP URL Format
- RFC 2256: A Summary of the X.500(96) User Schema for use with LDAPv3
- RFC 2829: Authentication Methods for LDAP
- RFC 2830: LDAPv3: Extension for Transport Layer Security
- RFC 3771: LDAP: Intermediate Response Message
- LDAP version 2 (LDAPv2) is a now Historic (see RFC 3494)

# 3. OpenLDAP telepítése és elindítása

Bizonyos lépéseket, beállításokat el kell végezni a címtár használata előtt. A legtöbb linuxos terjesztés esetén csomagból telepíthető a program, és néhány kérdés megválaszolása után használhatóvá is válik, azonban nem árt tisztában lenni vele, hogy hogyan lehet forrásból telepíteni, valamint elvégezni a címtár létrehozását.

Első lépésben bemutatjuk, hogyan telepíthetjük a programot forráskódból, majd pedig hogyan végezhető el a telepítés a Debian saját csomagkezelőjével.

## 3.1. Telepítés forrásból

A telepítéshez először is be kell szerezni a program forráskódját [tömörítve 3,6 MB], amelyet a program weboldaláról vagy tükörszervereiről tölthetünk le. A letöltés után jöhet a kicsomagolás:

```
$ cd /usr/src/
$ wget ftp://ftp.openldap.org/pub/OpenLDAP/\
    openldap-stable/openldap-stable-20060227.tgz
$ tar xfvz openldap-stable-20060227.tgz
$ cd openldap-2.3.20/
```

A fordítási folyamat ugyanaz, mint a legtöbb program esetén (`configure`, `make`, `make install`). A fordítás előkészítéséhez, előbb el kell végezni bizonyos beállításokat, amit a `configure` paranccsal tehetünk meg, de előbb nézzük meg a lehetőségeinket:

```
./configure --help
```

A telepítési könyvtárak beállításához adjuk meg a következő kapcsolókat a kívánt könyvtárral (elérési útvonalával együtt) `--prefix=pref`, `--exec-prefix=eprefix`, `--bindir=dir`. Ha nem adunk meg paramétert a `configure` szkriptnek, megpróbálja automatikusan megállapítani a szükséges beállításokat és előkészíti a telepítést az alapértelmezett helyre:

```
./configure
```

A kimenetet figyelve, megtudhatjuk, hogy a beállítások megfelelnek-e a kívánalmainknak. A

titkosított csatorna támogatásához (TSL/SSL) automatikusan ellenőrzi a megfelelő dev csomagok jelenlétét.

Ha minden rendben találtunk a `configure` futása után, akkor a fordítás előtt hozzuk létre a függőségeket, majd indítsuk el a fordítási folyamatot:

```
$ make depend
$ make
```

Sikeres fordítás esetén érdemes futtatni egy tesztet:

```
$ make test
```

Ha minden rendben, fejezzük be a bináris állományok telepítésével, amelyhez rendszergazdai jog kell:

```
$ su root -c 'make install'
```

### 3.1.1. A program elindítása

A program elindítása a bináris fájl futtatásával történik:

```
/usr/local/sbin/slapd [opciók]
```

Elinduláskor beolvassa a `/etc/ldap/slapd.conf` konfigurációs fájt, amit a „*-f fájl*” paraméterrel bírálhatunk felül. A konfigurációs fájl tesztelést a „*-t*” kapcsoló megadásával kérhetjük.

### 3.1.2. A program leállítása

Az LDAP szolgáltatást az alábbi paranccsal szüntethetjük meg:

```
kill -INT `cat /usr/local/var/slapd.pid`
```

Az ennél drasztikussabb leállítás adatvesztést és adatbázishibákat eredményezhet.

## 3.2. Telepítés Debian GNU/Linux operációs rendszeren

Kihasználva a rendszer adottságait és a csomagtelepítőjét, egyszerűen és kényelmesen juthatunk előre fordított, teljes funkcionalitású OpenLDAP kiszolgálóhoz. Adjuk ki az alábbi parancsot rendszergazdaként:

```
$ apt-get install slapd ldap-utils
```

A `slapd` csomag tartalmazza többek között a kiszolgálót, a többszörözőt (replikátort – `slurpd`) és pár alapvető programot. A segédprogramokat viszont az `ldap-utils` csomag foglalja magába, így érdemes azt is felrakni.

### 3.2.1. A program elindítása

Debian GNU/Linux rendszeren egyszerű dolgunk van, jól működő mechanizmus van a kezünkben a programok elindítására, újraindítására, leállítására. A program elindítása:

```
/etc/init.d/slapd start
```

A program elindítását szabályozhatjuk az alábbi fájl tartalmának módosításával is:

```
/etc/default/slapd
```

### 3.2.2. A program leállítása

A program leállítását szintén elvégezhetjük az init script segítségével:

```
/etc/init.d/slapd stop
```

# 4. Beállítás

A `slapd` szerverprogram konfigurációs fájlja a *slapd.conf* fájl, ennek beállításával tudjuk szabályozni a kiszolgáló működését. Azonban, hogy lokálisan is tudjuk módosítani az adatbázisban tárolt adatokat, a kliensoldali konfigurációs fájlt (*ldap.conf*) is be kell állítani.

## 4.1. Kiszolgáló beállítása

A kiszolgáló beállításához szerkesszük a */etc/ldap/slapd.conf* fájlt. A konfigurációs fájl több részre osztható. A könnyebb érhetőség kedvéért a különböző részeket külön tárgyaljuk.

### 4.1.1. Sémák

Minden bejegyzésnek tartalmaznia kell legalább egy „*objectclass*” tulajdonságot, ezen tulajdonságokat sémákban definiálják. Az engedélyezett sémákat külön meg kell adni:

```
# séma megadások
include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema

# séma ellenőrzés bekapcsolása
schemacheck     on
```

Ha bekapcsoljuk az ellenőrzést, akkor minden hozzáadás és módosítás alkalmával a szerver ellenőrzi, hogy a bejegyzés megfelel-e az objektum-osztályok definícióinak.

### 4.1.2. Futáshoz szükséges beállítások

Általános beállítások szerepelnek itt.

```
# pid fájl helye, az init script is itt keresi
pidfile         /var/run/slapd/slapd.pid

# szervernek átadott paraméterek listája
argsfile        /var/run/slapd.args

# naplózási szint
```

```
loglevel        0

# a betölthető modulok könyvtára
modulepath      /usr/lib/ldap
moduleload      back_bdb
```

### 4.1.3. Backend-re vonatkozó beállítások

A backend-re vonatkozó beállítások érvényesek lesznek a tartalmazott adatbázisokra, azonban az adatbázisoknál lehetőség van a felülbírálásra.

```
# backend típusának beállítása
backend         bdb

# ellenőrző pontok megadása <KB> <perc>
checkpoint      512 30
```

Az ellenőrző pont értelmezése az OpenLDAP 2.3-as változatban megváltozik. Előző verziókban csak akkor történik mentés, ha a *checkpoint* által megadott értékek valamelyike bekövetkezett és írási művelet is történt, 2.3-as verziótól kezdve, mindig történik mentés.

### 4.1.4. A keresési eredmények korlátozása

Nem megfelelően szűkített keresés nagyon sok bejegyzést adhat vissza. Lehetőségünk van arra, hogy maximalizáljuk a visszaadott eredmények darabszámát:

```
Sizelimit       100
```

### 4.1.5. Adatbázisra vonatkozó beállítások

Adatbázisra vonatkozó beállítások, amelyekkel a backend-hez tartozó beállításokat is felülbírálhatjuk.

```
# adatbázis beállítása
database        bdb

# utótag megadása, az LDAP-fa kiindulási pontja
suffix          "dc=proba,dc=hu"

# adatbázis fájlok tárolási könyvtára
directory       "/var/lib/ldap"
```

```
# indexelési szabályok
index           objectClass eq

# bejegyzés utolsó módosítási időpontjának mentése
lastmod         on

# többszörözési naplózás könyvtára az aktuális adatbázishoz
# replogfile    /var/lib/ldap/replog
```

**indexelés**

```
index {<attrlist> | default} [pres,eq,approx,sub,none]
```

Megadható értékek jelentése:

| **Megnevezés** | **jelentés** |
|---|---|
| attribute list (attrlist) | tulajdonságok |
| present (pres) | jelenlét |
| equality (eq) | egyenlőség |
| approximate (approx) | becslés |
| substring (sub) | szövegrész |
| none | indexelés kikapcsolása |

Alapértelmezés szerint nem készül indexelés, de mindenképpen érdemes bekapcsolni, legalább *objectClass* szerinti *equality* (egyenlőség) indexelést.

```
index objectClass eq
```

### 4.1.6. TLS kapcsolat biztosítása

Egyszerű azonosítás esetén (simple authentication) érdemes magát a kapcsolatot titkosítani, használjunk TLS-t:

```
TLSCertificateFile      /etc/ssl/private/ldap.req.pem
TLSCertificateKeyFile   /etc/ssl/private/ldap.key.pem
TLSCACertificateFile    /etc/ssl/private/ldap.req.pem
```

Szükséges fájlok előállítása:

```
openssl req -new -x509 -nodes -out req.pem -keyout key.pem -days 3650
```

## 4.1.7. Jogosultsági beállítások

A különböző hozzáférési jogokat egyszerűen, de nagyon hatékonyan tudjuk szabályozni. Az alap séma a következő:

```
access to <mihez>[.hatókör] [szűrő=<ldap szűrő>]
     by <kinek> <mit>
```

#### 4.1.7.1. „<mihez>[.hatókör]” elem értékei

A „*<mihez>[.hatókör]*” használatát az OpenLDAP dokumentumban leírt példán keresztül könnyen megérthetjük:

| szám | az adatbázis különböző bejegyzései |
|---:|---|
| 0 | dc=proba |
| 1 | cn=admin,dc=proba |
| 2 | ou=people,dc=proba |
| 3 | uid=proba,ou=people,dc=proba |
| 4 | cn=addresses,uid=proba,ou=people,dc=proba |
| 5 | uid=pelda,ou=people,dc=proba |

A különböző hatókörök hatására az illeszkedések a következőképpen alakulnak:

| <mihez>[.hatókör] | egyezés |
|---|---|
| dn.base="ou=people,dc=proba" | 2 |
| dn.one="ou=people,dc=proba" | 3, 5 |
| dn.subtree="ou=people,dc=proba" | 2, 3, 4, 5 |
| dn.children="ou=people,dc=proba" | 3, 4, 5 |

A „*[szűrő=<ldap szűrő>]*” megadásával tovább tudjuk szűrni a „*<mihez>[.hatókör]*” illeszkedését:

```
to dn.one="ou=users,dc=proba,dc=hu” filter=(objectClass=posixAccount)
```

Melynek következtében csakis azokra a bejegyzésekre fog illeszkedni, melyek a megadott *dn* alá tartoznak közvetlenül és unix account-ok.

#### 4.1.7.2. <kinek> elem értékei

A „<*kinek*>” elem által felvehető értékeket és jelentésüket az alábbi táblázat foglalja össze:

| **speciális elem** | **jelentése** |
|---|---|
| * | mindenki (authenticarted és non-authenticated) |
| anonymous | anonymous (non-authenticated) |
| users | azonosított (authenticarted) felhasználó |
| self | megegyezik a célbejegyzéssel |
| dn=<regex> | szabályos kifejezésre illeszkedő bejegyzések |
| dn.<hatókör>=<dn> | <dn> és hatókörén belül eső bejegyzések |

#### 4.1.7.3. <mit> elem értékei

A kiadható jogok, és jelentésüket az alábbi táblázat foglalja össze:

| **jog** | **leírás** |
|---|---|
| none | hozzáférés megtagadva |
| auth | szükséges az azonosításhoz |
| compare | szükséges az összehasonlításhoz |
| search | szükséges a kereséshez |
| read | szükséges a keresési találatok olvasásához |
| write | szükséges a módosításhoz, átnevezéshez |

#### 4.1.7.4. Példák

Az előbbi lehetőségeket nézzük át újra, az OpenLDAP dokumentációban található példákon keresztül.

Mindenki számára megengedve az olvasás:

```
access to *
```

```
     by * read
```

A következő példában megengedjünk mindenkinek számára, hogy saját magát írhassa, anonymous számára az azonosítást, a többiek számára pedig az olvasást.

```
access to *
     by self write
     by anonymous auth
     by * read
```

A jogok feldolgozása sorfolytonos, így az utolsó sorban levő „*” (csillag) helyére írhattunk volt „*users*”-t is, mert ebben az esetben ugyanazt a hatást fejtik ki.

Az alábbi példa tökéletes arra, hogy bemutassuk, igen is fontos a jogok megadásának sorrendje:

```
access to dn.children="dc=proba,dc=hu"
     by * search
access to dn.children="dc=hu"
     by * read
```

A „*dc=proba,dc=hu*” ágban megengedjük a keresést, a „*dc=hu*” ágban pedig az olvasást. Vegyük észre, hogy ha a két jog megadását felcseréltük volna, akkor sosem lenne illeszkedés a „*dc=proba,dc=hu*” DN-re, mivel ő maga is a „*dc=hu*” ágban tartozik, így a „*dc=proba,dc=hu*” ágba tartozó elemek is kapnak olvasási jogok a csak keresési jog helyett.

## 4.2. Kliensoldali beállítás

A kliensoldali beállításhoz szerkesszük a */etc/ldap/ldap.conf* fájlt.

```
BASE            dc=proba,dc=hu
URI             ldap://ldap.proba.hu
TLS_CACERT      /etc/ssl/private/ldap.pem
```

A sorok jelentése sorrendben: az első sorban a kiindulási pontot adjuk meg a fában, a másodikban a szerver címét. A harmadik sor a TLS kapcsolathoz szükséges CaCert állományt és útvonalát tartalmazza.

# 5. Sémák

A *slapd.conf* fájlban sémát az *attributeType* és *objectClass* használatával is megadhatunk, de külön fájlban tárol sémadefiníciók is beemelhetők a már említett „*include <sémafájl>*” megadásával. Debian rendszer alatt a különböző sémákat definiáló fájlok a `/etc/ldap/schema` könyvtárban találhatók.

## 5.1. Új séma létrehozása

Új séma létrehozása előtt igényelnünk kell egy egyedi *OID prefixre*., amelyet a megfelelő regisztrációs intézettől lehet igényelni. Egyik legismertebb ingyenes regisztrátor az Internet Assigned Numbers Authority[2]. Az új elemek létrehozásának szabályait betartva tetszőleges számú osztály és attribútum definiálható.

### 5.1.1. Object Identifiers

Séma létrehozásakor először létrehozzuk a tulajdonságokat (attribútumokat), megadjuk a nevét és szintaxisát, milyen illesztés végezhető rá, egyértékű vagy több, stb. Szülő attribútum megadásakor átvesz minden tulajdonságot. Ezen felül a szülőn alkalmazott szűrők vizsgálatánál a gyerek attribútum értéke is ellenőrzésre kerül.

Ezután objektum típusokat állítunk elő az attribútumokból, mely során meg kell adnunk, hogy mely tulajdonságok a *kötelezők* és melyek a *megadhatók*. Itt is van öröklődés.

Mindegyik objektum, attribútum és szintaxis rendelkezik egyedi azonosítóval (OID). Ilyen azonosítót bárki ingyen igényelhet a http://www.iana.org/cgi-bin/enterprise.pl weboldalon. Az ingyenesség célja, hogy az azonosítók ne ütközzenek. Mind az IP-címeknél, itt is van egy lokális tartomány, amelyet nem adnak ki, mindenki csak helyi használatra veheti igénybe. Ez a hálózat az 1.1-es.

Az OID az ITU-T X.208 (ASN.1) ajánlás leszármazottja. Az OID-ek összessége egy faszerű struktúrába rendezett adathalmaz. A legfelső szinttől kezdve végigjárható a http://www.alvestrand.no/objectid/top.html weboldalon, amely alapján megtalálhatjuk az Internet

2 http://www.iana.org

OID számát is:

| **OID** | **megnevezés** |
|---|---|
| 1 | ISO assigned |
| 1.3 | ISO Identified Organization |
| 1.3.6 | US Department of Defense |
| 1.3.6.1 | OID assignments from 1.3.6.1 - Internet |
| 1.3.6.1.4 | Internet Private |
| 1.3.6.1.4.1 | IANA-registered Private Enterprises |

| **OID** | **megnevezés** |
|---|---|
| 1.3.6.1.1 | Directory |
| 1.3.6.1.2 | Management (mgmt) |
| 1.3.6.1.3 | Experimental |
| **1.3.6.1.4** | **Internet Private** |
| 1.3.6.1.5 | Security |
| 1.3.6.1.6 | SNMPv2 |
| 1.3.6.1.7 | mail |

| **OID** | **megnevezés** |
|---|---|
| 1.3.6.1.4 | Internet Private |
| 1.3.6.1.4.1 | IANA-registered Private Enterprises |
| 1.3.6.1.4.1.1466 | Mark Wahl (Critical Angle) |
| 1.3.6.1.4.1.1466.115 | LDAPv3 Schema Framework (Syntaxes) |
| 1.3.6.1.4.1.1466.115.112 | LDAPv3 Syntaxes |
| 1.3.6.1.4.1.1466.115.112.1 | LDAPv3 Syntaxes |
| 1.3.6.1.4.1.1466.115.112.1.x | http://www.alvestrand.no/objectid/1.3.6.1.4.1.1466.115.121.1.html |

## 5.2. Pár objectClass és attributetype példa

Érdemes átnézni pár *objectClass* és *attributetype* típus definíciót.

NIS.schema fájlból a PosixAccount objectClass:

```
objectclass ( 1.3.6.1.1.1.2.0 NAME 'posixAccount' SUP top AUXILIARY
     DESC 'Abstraction of an account with POSIX attributes'
     MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
     MAY ( userPassword $ loginShell $ gecos $ description ) )
```

CORE.scheme fájlból a Person objectClass:

```
objectclass ( 2.5.6.6 NAME 'person'
     DESC 'RFC2256: a person'
     SUP top STRUCTURAL
     MUST ( sn $ cn )
     MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )
```

Attribútum típusra egy példa:

```
attributetype ( 2.5.4.31 NAME 'member'
     DESC 'RFC2256: member of a group'
     SUP distinguishedName )
```

# 6. Adatbázis létrehozása

Az adatbázist két módon hozhatjuk létre: *on-line* és *off-line*. Az on-line módon történő adatbázis létrehozása esetén fut az LDAP kiszolgáló és egy LDAP kliens segítségével elvégezzük a szükséges lépéseket. Néhány száz, esetleg ezer elem esetén javasolt. Több ezer elem feltöltése esetén az off-line módszer ajánlott, azonban speciális segédprogramok szükségesek, amelyek nem támogatják mindegyik adatbázis-motort.

Debian GNU/Linux rendszeren csomagból való telepítés esetén az alapbeállításokat megkérdezi a rendszer, majd létrehozza az adatbázist, így nem kell elvégeznünk kézzel, azonban érdemes tudni a létrehozás lépéseit.

## 6.1. on-line

Ahogy a neve is sugallja, úgy hozzuk létre az adatbázist, hogy a `slapd` program fut. Választunk egy LDAP kliens programok (jelen esetben: `ldapadd`) és hozzáadjuk vele az adatbázis alapját képező elemeket. Egyszer kell csak megtenni, de bizonyosodjunk meg róla, hogy az alábbi beállításokat elvégeztük a konfigurációs fájlban (*slapd.conf*) a `slapd` elindítása előtt:

```
# LDAP-fa kiindulási pontja.
suffix "dc=pelda,dc=hu"

# adatbázisfájlok helye (legyen joga a slapd programnak írni)
directory /var/lib/ldap

# super-user beállítása, addig kell, amíg létrehozzuk az adatbázist
# csak hogy feloldjuk a tyúk-meg-a-tojás problémát...
rootdn "cn=admin,dc=pelda,dc=hu"
rootpw jelszó

# indexelési beállítások, nem kötelezőek, de növelik a hatékonyságot
index cn,sn,uid pres,eq,approx,sub
index objectClass eq
```

A beállítások elvégzése után indítsuk el a `slapd` programot, majd hozzuk létre az LDAP-fát és az *admin* felhasználót. Hozzunk létre egy szöveges fájlt (*proba.ldif*) és írjuk bele az alábbi adatokat:

```
# Próba cég LDAP-fájának kiinduló pontja
dn: dc=proba,dc=hu
objectClass: top
objectClass: dcObject
objectClass: organization
o: Proba Ceg
dc: proba

# LDAP adatbázis admin felhasználója
dn: cn=admin,dc=proba,dc=hu
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: {SSHA}ojCFfGBId0u8z95gqkKl6tNE14RaLrBF
```

A *userPassword* értékét a „`slappasswd -h {SSHA}`” paranccsal állítottuk elő („xxx” jelszóval). A „*-h <séma>*” paraméterrel adható meg a kódolási eljárás, amely a következő lehet: „{CRYPT}”, „{MD5}”, „{SMD5}”, „{SSHA}”, valamint „{SHA}”. Kódolatlan formában is tárolhatjuk a jelszavakat, de ez kerülendő:

```
userPassword:: {CLEARTEXT}jelszó
```

Ezután adjuk hozzá a *proba.ldif* fájl tartalmát az adatbázishoz az `ldapadd` klienssel:

```
$ ldapadd -x -W -D "cn=admin,dc=proba,dc=hu" -f proba.ldif
```

A parancs után bekéri a *slapd.conf* fájlban beállított jelszót. Ha sikeresen megtörtént az adatbázis frissítése, akkor vegyük ki a „*rootdn*” és „*rootpw*” kezdetű sorokat a *slapd.conf* fájlból, indítjuk újra a szolgáltatást és élvezzük a használatra felkészített LDAP kiszolgálónkat.

## 6.2. off-line

A módszer alkalmazása előtt mindenképpen bizonyosodjunk meg róla, hogy nem fut az LDAP szolgáltatás mielőtt nekiállnák az adatbázis feltöltésének.

A művelet során a `slapadd` parancsot fogjuk használni:

```
$ slapadd -l <adatfájl> [-n <sorszám> | -b <suffix>]
```

Az adatbázishoz adandó elemeket tartalmazó fájlt a „*-l <adatfájl>*” paraméterrel adhatjuk meg.

Az elemek az alapértelmezett adatbázishoz adódnak hozzá, amit felülbírálhatunk a „*-n <adatbázis sorszáma>*” vagy a „*-b <suffix>*” paraméterekkel. Az utóbbi két paraméter használata kizárja egymást, egyszerre csak az egyik adható meg.

### 6.2.1. Adatbázis újraindexelése

Néha szükséges lehet újraépíteni az adatbázis indextábláját, különösen, ha módosítjuk a *slapd.conf* állomány ide vonatkozó részét. Indexek készítését a `slapindex` programmal tehetjük meg:

```
$ slapindex [-n <sorszám> | -b <suffix>]
```

A „*-n*” és „*-b*” paraméterek jelentése megegyezik a `slapadd` paramétereinek jelentésével

### 6.2.2. Adatbázis mentése fájlba

A `slapcat` program segítségével lehet az adatbázist „dump-olni”. Ami hasznos lehet, ha egy ember által is olvasható mentést akarunk készíteni az adatbázisról, vagy off-line akarjuk szerkeszteni az adatbázist. A használata szinte megegyezik a `slapadd` programéval:

```
$ slapcat -l <fájlnév> [-n <sorszám> | -b <suffix>]
```

A „*-n*” és „*-b*” paraméterek jelentése megegyezik a `slapadd` paramétereinek jelentésével. A mentés alapesetben az alapértelmezett kimenetre történik (standart output), de a „*-l <fájlnév>*” paraméterrel fájlba is írathatjuk.

## 6.3. LDIF-fájl formátuma

Az LDIF egy olyan szöveges fájl, amely LDAP elemeket, műveletek tartalmaz. Felépítése:

```
# megjegyzés
dn: <egyedi azonosító>
<kulcs>: <érték>
<kulcs>: <érték>

...
```

Kettős kereszttel („#”) kezdődő sorok megjegyzésként értelmeződnek. A *<kulcs>* lehet egyszerű,

úgymint „*cn*”, „*objectClass*” vagy „*1.2.3*” (attribútum-típussal azonosított OID) vagy opciókat is tartalmazhat, például:. cn;lang_en_US vagy userCertificate;binary.

Egy bejegyzést több sorba törve is írhatunk, ha bevezetjük egy szóközzel vagy egy tabulátorral, így ez a példa:

```
dn: cn=Jonas Zsolt,dc=proba,dc=
 hu
cn: Jonas
     Zsolt
```

értelmezés szerint teljesen megegyezik ezzel:

```
dn: cn=Jonas Zsolt,dc=proba,dc=hu
cn: Jonas Zsolt
```

Egy <*kulcs*>-hoz több értéket is rendelhetünk külön sorban megadva:

```
cn: Jonas Zsolt
cn: Zsolt Jonas
```

Ha az <*érték*> tartalmaz nem nyomtatható karaktert, vagy szóközzel, kettősponttal, esetleg 'kisebb mint' jellel ('<') kezdődik, akkor a <*kulcs*>-ot egy helyett kettő kettőspont követ és Base64 kódolással történik a tárolása. ' szóközzel kezdve' tárolása így néz ki:

```
cn:: IHN682v2enplbCBrZXpkdmU=
```

Több LDAP elemet is tartalmazhat egy LDIF fájl. Az elemeket üres sorral kell elválasztani egymástól, ahogy a példa is mutatja:

```
# Jonas Zsolt
dn: cn=Jonas Zsolt,dc=proba,dc=hu
objectClass: person
cn: Jonas Zsolt
cn: Zsolt Jonas
sn: Jonas
# Base64 kódolású tartalom
jpegPhoto:: /9j/4AAQSkZJRgABAgAAZABkAAD/7AARRHVja3kAA...
TAAAAAAQMAFQQDBgoNAAAJygAADZ0AABcwAAAm5v/bAIQABgQEBAU...
KcwoKDBAMDAwMDAwQDA4PEA8ODBMTFBQTExwbGxscHx8fHx8fHx8f...

# Sinko Gergely
```

```
dn: cn=Sinko Gergely,dc=proba,dc=hu
objectClass: person
cn: Sinko Gergely
sn: Sinko
# Kép fájlból
jpegPhoto:< file:///útvonal/a/kép/fájlhoz/kep.jpeg.base64
```

Vegyük észre, hogy van olyan <*kulcs*>, amelynek <*érték*>-ét többféleképpen is megadhatjuk. Bináris adat tárolása Base64 kódolással történik, amelyet beírhatunk közvetlenül is a fájlba, de megadhatunk egy külső fájlt is forrásnak. Arra azonban figyeljünk, hogy külső fájl megadásakor rakjuk ki a 'kisebb mint' jelet ('<') különben nem a fájl tartalma lesz az <*érték*>*,* hanem a fájl neve az útvonalával együtt.

Base64 kódolású tartalom előállításához telepítsük fel a *mime-codecs* csomagot:

```
$ apt-get install mime-codecs
```

Ahhoz, hogy az adatbázisba írhassuk a bináris adatot, át kell konvertálni Base64 kódolásúvá:

```
$ base64-encode < kep.jpeg > kep.jpeg.base64
```

Még egy apró megjegyzés: a sorvégi szóközök az <*érték*> részét képezik, nincs automatikus eltávolítás, se a többszörös előfordulás esetén.

# 7. Adatbázis on-line módosítása

Több segédprogram is rendelkezésünkre áll, hogy különböző módosításokat végezhessünk az adatbázison. Hozhatunk létre új LDAP elemet, vagy egy már meglévőt módosíthatunk, törölhetünk. Az alábbi segédprogramok találhatók az *ldap-utils* csomagban:

- ldapadd, ldapmodify
- ldapmodrdn
- ldappasswd
- ldapdelete
- ldapsearch
- ldapcompare

A segédprogramok kapcsolói jelentésikben megegyeznek, ezért érdemes az elején átnézni őket:

| **paraméter** | **jelentés** |
|---|---|
| -v | bőbeszédű mód bekapcsolása |
| -x | egyszerű azonosítás (authentikáció, ne használja a SASL azonosítást) |
| -D <dn> | milyen néven (DN) kívánunk az LDAP címtárat használni |
| -W | a névhez tartozó jelszó (helyette használható a „*-w jelszó*” is) |
| -f <fájl> | előre megírt LDIF fájl használata forrásnak |

Előfordulhat, hogy valamelyik segédprogramnál használunk más paramétert is, de azokat majd ott helyben értelmezzük.

## 7.1. Hozzáadás, módosítás

Elemet hozzáadni, módosítani az `ldapmodify` programmal lehet, az `ldapadd` csak egy *hardlink* az `ldapmodify` programra.

Adjunk hozzá a „*Jónás Zsolt*” embert a címtárhoz. Hozzuk létre a *proba.uj.ldif* fájt az alábbi

tartalommal:

```
dn: cn=Jonas Zsolt,dc=proba,dc=hu
objectclass: inetorgperson
cn: Jonas Zsolt
sn: Jonas
displayname: jonci
homephone: 555-555
```

Majd használjuk az `ldapmodify` parancsot:

```
$ ldapmodify -a -x -W -D "cn=admin,dc=proba,dc=hu" -f proba.uj.ldif
Enter LDAP Password: <admin felhasználó jelszava>
add objectclass:
        inetorgperson
add cn:
        Jonas Zsolt
add sn:
        Jonas
add displayname:
        jonci
add homephone:
        555-555
adding new entry "cn=Jonas Zsolt,dc=proba,dc=hu"
modify complete

$
```

Módosítsuk a bejegyzés tulajdonságait: állítsunk be neki e-mail címet, változtassuk meg a kijelzett nevét, valamint töröljük a telefonszámát. Ehhez hozzuk létre a „*proba.mod.ldif*” fájlt az alábbi tartalommal:

```
dn: cn=Jonas Zsolt,dc=proba,dc=hu
changetype: modify
add: mail
mail: jonci@proba.hu
-
replace: displayname
displayname: Jonas Zsolt
-
delete: homephone
-
```

Változtassuk meg az adatokat az `ldapmodify` paranccsal:

```
$ ldapmodify -x -W -D "cn=admin,dc=proba,dc=hu" -f proba.mod.ldif
```

```
Enter LDAP Password: <admin felhasználó jelszava>
add mail:
       jonci@proba.hu
replace displayname:
        Jonas Zsolt
delete homephone:
modifying entry "cn=Jonas Zsolt,dc=proba,dc=hu"
modify complete

$
```

## 7.2. Relatív név (RDN) módosítása

Egy bejegyzés *RDN* tulajdonságának megváltoztatásához az `ldapmodrdn` programot használhatjuk. A használatával nem helyezhetünk át egy bejegyzést a fában, mivel csak az *RDN* értékét tudjuk módosítani, a szülő *DN* értékét nem. A „`-r`” kapcsoló megadásával a program törli a régi *cn* tulajdonságot. Ezek után lássunk példát az *RDN* érték módosítására:

```
$ ldapmodrdn -r -x -W -D "cn=admin,dc=proba,dc=hu" \
     "cn=Jonas Zsolt,dc=proba,dc=hu" "cn=I. Jonas Zsolt"
$
```

## 7.3. Jelszócsere

A „*-S*” kapcsolóval kérhetjük be az új jelszót, a „*-s jelszó*” paraméterrel közvetlenül is megadhatjuk. Ha egyiket se használjuk, akkor egy véletlenül előállított jelszót kaput, amit a képernyőre írva tudat velünk.

Saját jelszavunk megváltoztatása jelszó megadása nélkül, majd jelszó megadásával:

```
$ ldappassword -x -W -D "cn=admin,dc=proba,dc=hu" -S
New password: <adjuk meg a saját új jelszavunk>
Enter LDAP Password: <admin felhasználó jelszava>
$
```

A „*cn=proba,dc=proba,dc=hu*” elem jelszavának megváltoztatása:

```
ldappasswd -x -W -D "cn=admin,dc=proba,dc=hu" -S \
     "cn=Jonas Zsolt,dc=proba,dc=hu"
New password: <adjuk meg a Jonas Zsolt új jelszavát>
Re-enter new password: <biztos, ami biztos alapon még egyszer>
```

```
Enter LDAP Password: <admin felhasználó jelszava>
Result: Success (0)
$
```

## 7.4. Keresés

Kereséskor meg kell adni a *kiindulópontot* (*base*) és a keresés *hatókörét* (*scope*). További szűkítést is megadhatunk *szűrők* (*filter*) alkalmazásával.

#### 7.4.1.1. base

Egy olyan DN, ahonnan kell kezdeni a keresést. A fában a kiindulópontnál feljebb levő elemek nem szerepelnek a keresésesben.

#### 7.4.1.2. scope

Háromféle hatókört különböztetünk meg:

- `base`: csak a base DN által megadott elem szerepel a hatókörben
- `one`: a base DN által meghatározott elem és az egy szinttel alatta levő elemek szerepelnek a keresésben
- `sub`: a base DN által meghatározott teljes részfa szerepel a keresésben

#### 7.4.1.3. filter

Keresés során megkapjuk az összes elemet, amelyet a *base* és a *scope* meghatároz, azonban van mód a keresés eredményét tovább szűkíteni *szűrők* megadásával. Meghatározhatjuk, hogy csak akkor legyen illeszkedés egy elemre, ha az elem valamely tulajdonságának értéke megegyezik vagy tartalmazza az általunk megadott adatot. Szöveg kezelése nem nagybetű-érzékeny (non-case sensitive).

A szűrésnél használható operátorok:

| operátor | jelentése |
|---|---|
| & | és |
| \| | vagy |
| ! | tagadás |
| ~= | majdnem azonos |
| >= | nagyobb vagy egyenlő |
| <= | kisebb vagy egyenlő |
| * | bármi |

Lássunk pár példát:

```
# objectclass-nak „posixAccount”-nak kell lennie
(objectclass=posixAccount)

# cn érékének „Jonas”-sal kell kezdődnie
(cn=Jonas*)

# uid értéke „jonas” vagy „jonci” lehet
(|(uid=jonas)(uid=jonci))

# uid értéke vagy „jonas” vagy „jonci” lehet, és emellett az
# objectclass-nak „posixAccount”-nak kell lennie
(&(|(uid=jonas)(uid=jonci))(objectclass=posixAccount))

# objectclass nem lehet inetOrgPerson
(!(objectclass=inetOrgPerson))

# objectclass-nak „person”-nak kell lennie,
# miközben cn nem kezdődhet „Jonas”-sal
(&(objectclass=person)(!(cn=Jonas*)))
```

#### 7.4.1.4. attribute

A keresés eredményét még ennél is jobban szűrhetjük azzal, hogy a keresésre illeszkedő elemek nem minden tulajdonságát listáztatjuk ki, csak a számunkra szükségeseket. Egy elemhez sok-sok tulajdonság is tartozhat, azonban minek íródjon ki mind, ha mi csak adott tulajdonságok értékeire vagyunk kíváncsiak.

Ezen után lássunk egy példát, amiben a keresés a „*dc=proba,dc=hu*” szintről kezdődik, csak egy szinttel lejjebb tart csak, és a *cn* tulajdonság értékének elejének „*Jonas*”-nak kell lennie. Az erre illeszkedő elemeknek csak a *cn* és *mail* tulajdonságaik jelennek meg – értékükkel együtt – a

kimeneten.

```
$ ldapsearch -x -W -D "cn=admin,dc=proba,dc=hu" \
     -b "dc=proba,dc=hu" -s one "(cn=Jonas*)" 'cn' 'mail'
Enter LDAP Password: <admin felhasználó jelszava>
filter: (cn=Jonas*)
requesting: cn mail
# Jonas Zsolt, proba.hu
cn: Jonas Zsolt
mail: jonci@proba.hu

...
$
```

## 7.5. Összehasonlítás

Van mód egy bejegyzés valamely értékét összehasonlítani egy általunk meghatározott tulajdonság feltételezett értékével. Az összehasonlítás után TRUE (igaz) értéket kapunk, ha az értékek megegyeznek, és FALSE (hamis) értéket, ha nincs egyezés:

```
$ ldapcompare -x-W -D "cn=admin,dc=proba,dc=hu" \
     "cn=Jonas Zsolt,dc=proba,dc=hu" "cn: Jonas Zsolt"
Enter LDAP Password: <admin felhasználó jelszava>
DN:cn=Jonas Zsolt,dc=proba,dc=hu, attr:cn, value:Jonas Peter
FALSE
$ ldapcompare -x-W -D "cn=admin,dc=proba,dc=hu" \
     "cn=Jonas Zsolt,dc=proba,dc=hu" "cn: Jonas Zsolt"
Enter LDAP Password: <admin felhasználó jelszava>
DN:cn=Jonas Zsolt,dc=proba,dc=hu, attr:cn, value:Jonas Zsolt
TRUE
$
```

## 7.6. Törlés

Mindenezek után töröljük a bejegyzést a címtárból, ehhez a „*proba.torol.ldif*” fájlt az alábbi tartalommal hozzuk létre:

```
cn=Jonas Zsolt,dc=proba,dc=hu
```

Nem tévedés, a „*dn:* ” tagot nem szabad kitenni a sor elejére! Majd, használjuk az `ldapdelete` programot a törléshez:

```
$ ldapdelete -x -W -D "cn=admin,dc=proba,dc=hu" -f proba.torol.ldif
Enter LDAP Password: <admin felhasználó jelszava>
cn=Jonas Zsolt,dc=proba,dc=hu
deleting entry "cn=Jonas Zsolt,dc=proba,dc=hu"
Delete Result: Success (0)
$
```

## 7.7. LDIF fájl használata nélkül

Úgy is tudjuk módosítani az adatbázist, ha a parancsokat előtte nem írjuk LDIF fájlba. Ha elhagyjuk a „*-f ldiffájl*” paramétert, akkor a jelszó megadása után a használt segédprogram az alapértelmezett bemeneten várja a parancsokat. Annyi különbség van, hogy parancsokat a „*fájl vége*” (*EOF*) jellel kell lezárni:

```
$ ldapmodify -x -W -D "cn=admin,dc=proba,dc=hu"
Enter LDAP Password: <admin felhasználó jelszava>
dn: cn=Jonas Zsolt,dc=proba,dc=hu
changetype: modify
replace: mail
mail: janos@proba.hu
-
delete: jpegphoto
^d
dn: cn=Jonas Zsolt,dc=proba,dc=hu
changetype: modify
delete: mail

^d
^d
$
```

Érdemes megjegyezni, hogy a fájl vége (EOF) jel platformfüggő:

- Control-D (^d) a legtöbb UNIX rendszeren
- Control-Z (^z) majd ENTER a Windows rendszereket

# 8. Többszörözés

TODO

# 9. Rendszerbeállítás kliens oldalon

Miután létrehoztunk egy LDAP címtár szolgáltatást, már csak a rendszerünket kell felkészíteni annak a használatához.

## 9.1. Névszolgáltatás használata

Legelső teendők, hogy képessé tesszük rendszerünket, hogy használja az LDAP címtárat, ehhez tegyük fel a `libnss-ldap` csomagot:

```
$ apt-get install libnss-ldap
```

Ezután módosítsuk a */etc/nsswitch.conf* fájl tartalmát úgy, az LDAP címtárat is használja a névfeloldáshoz:

```
passwd:     files ldap
group:      files ldap
shadow:     files ldap
```

Majd nincs más hátra, állítsuk be az LDAP címtárra jellemző tulajdonságokat a */etc/libnss-ldap.conf* fájlban:

```
# host 127.0.0.1
uri ldap://ldap.proba.hu/ ldap://ldap-backup.proba.hu/
base dc=proba, dc=hu
scope sub
pam_filter objectclass=posixAccount
pam_password crypt
ssl start_tls
```

## 9.2. Azonosítás beállítása

Sikeresen elvégzett módosítások után készítsük fel a PAM (*Pluggable Authentication Modules*) szolgáltatást az LDAP címtárból történő azonosításra, ehhez azonban előtte telepíteni kell a *libpam-ldap* csomagot:

```
$ apt-get install libpam-ldap
```

PAM beállítása a `/etc/pam_ldap.conf` fájl szerkesztésével:

```
uri ldap://ldap.proba.hu/
base dc=proba,dc=hu
scope sub
pam_filter objectclass=posixAccount
pam_password crypt
ssl start_tls
```

**/etc/pam.d/common-account**

```
account sufficient       pam_ldap.so
account required         pam_unix.so try_first_pass
```

**/etc/pam.d/common-auth**

```
auth sufficient          pam_ldap.so
auth required            pam_unix.so try_first_pass
```

Az azonosításhoz rendelhetünk szűrést is. Például csak a 10000-es csoportba tartozók engedése:

```
auth sufficient          pam_ldap.so filter=(gidnumber=10000)
```

**/etc/pam.d/common-password**

```
password sufficient      pam_ldap.so
password required        pam_unix.so try_first_pass
```

**/etc/pam.d/common-session**

```
session sufficient       pam_ldap.so
session required         pam_unix.so
```

## 9.3. Névszolgáltatás cache-elése

Névfeloldás során elég sok kérést intézhet a rendszer a címtárhoz. Érdemes lehet egy gyorsítótárat üzembe helyezni, hogy csökkentsük a címtár terhelését, de jó szolgálatot tesz akkor is, ha a címtár és a kliens nem egy számítógépen van. Különösebb beállítást nem igényel, egyszerűen csak telepítsük fel:

```
apt-get install nscd
```

## 9.4. Ellenőrzés

Az ellenőrzéshez hozzunk létre egy teszt környezetet. Létrehozzuk a szükséges szervezeti egységeket és egy-egy felhasználót és csoportot:

Az *LDIF_fájl* tartalma:

```
# Felhasználok szervezeti egység létrehozása
dn: ou=Users,dc=proba,dc=hu
objectClass: organizationalUnit
ou: Users

# Csoportok szervezeti egység létrehozása
dn: ou=Groups,dc=proba,dc=hu
objectClass: organizationalUnit
ou: Groups

# users csoport létrehozása
dn: cn=ldapusers, ou=Groups,dc=proba,dc=hu
objectClass: posixGroup
cn: ldapusers
gidNumber: 10000
description: Posix users

# 'Jonas Zsolt' felhasználó létrehozása
dn: cn=Jonas Zsolt,ou=Users,dc=proba,dc=hu
objectclass: posixaccount
cn: Jonas Zsolt
gecos: Jonas Zsolt
uid: jonci
uidnumber: 10000
gidnumber: 10000
homedirectory: /srv/home/jonci
loginshell: /bin/bash
userpassword:: proba
```

Ezek után adjuk hozzá az elemeket a címtárhoz:

```
$ ldapmodify -a -x -W -D "cn=admin,dc=proba,dc=hu" -f LDIF_fájl
Enter LDAP Password: <admin felhasználó jelszava>
$
```

A hozzáadás után ellenőrizzük a felhasználó meglétét:

```
$ getent passwd
jonci:x:1000:1000:,,,:/home/jonci:/bin/bash
...
```

```
proba:x:10000:10000:,,,:/srv/home/proba:/bin/bash
...
$
```

Teszteljük le a helyes működést egy fájl létrehozásával:

```
$ touch /tmp/teszt
$ chmod 10000:10000 /tmp/teszt
$ ls -l /tmp/teszt
-rw-r--r--    1 jonci   ldapusers        0 Apr 28 22:32 /tmp/teszt
$
```

Végül próbáljuk ki a kihagyhatatlan `id` programot is:

```
$ su jonci
$ id
uid=10000(jonci) gid=10000(ldapusers) groups=10000(ldapusers)
$
```

# Irodalomjegyzék

[1] – OpenLDAP kézikönyvek (http://www.openldap.org/doc/)

[2] – LDAP HOGYAN – http://tldp.fsf.hu/HOWTO/LDAP-HOWTO-hu/

[3] – LDAP pár szóban – http://padre.web.elte.hu/ldap.html

[4] – Google ☺ – http://www.google.com