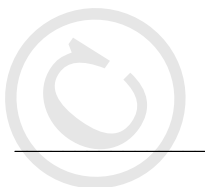


Student-notities
LDAP directory service
Voorbeeld-hoofdstuk

03. Attributen en filters

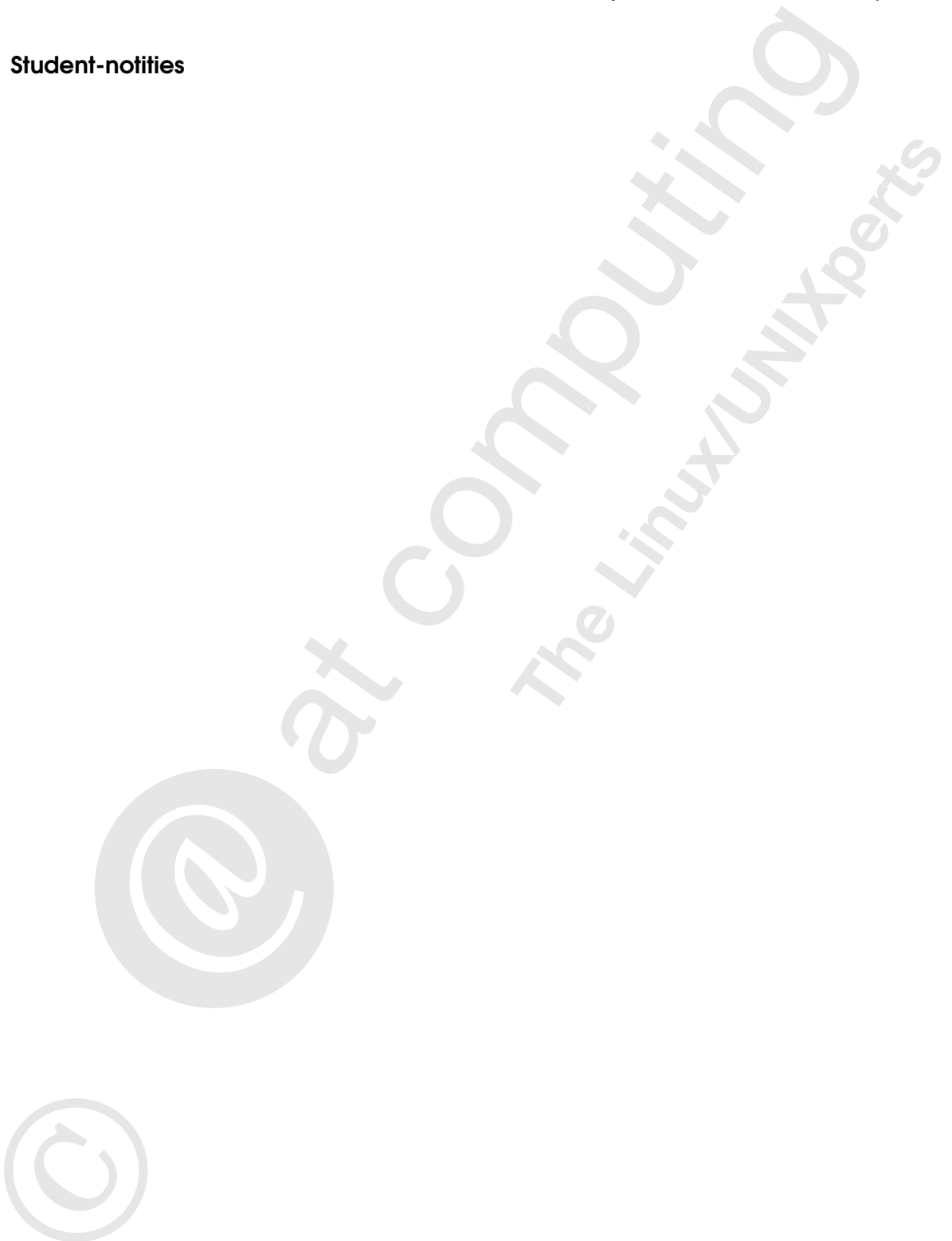


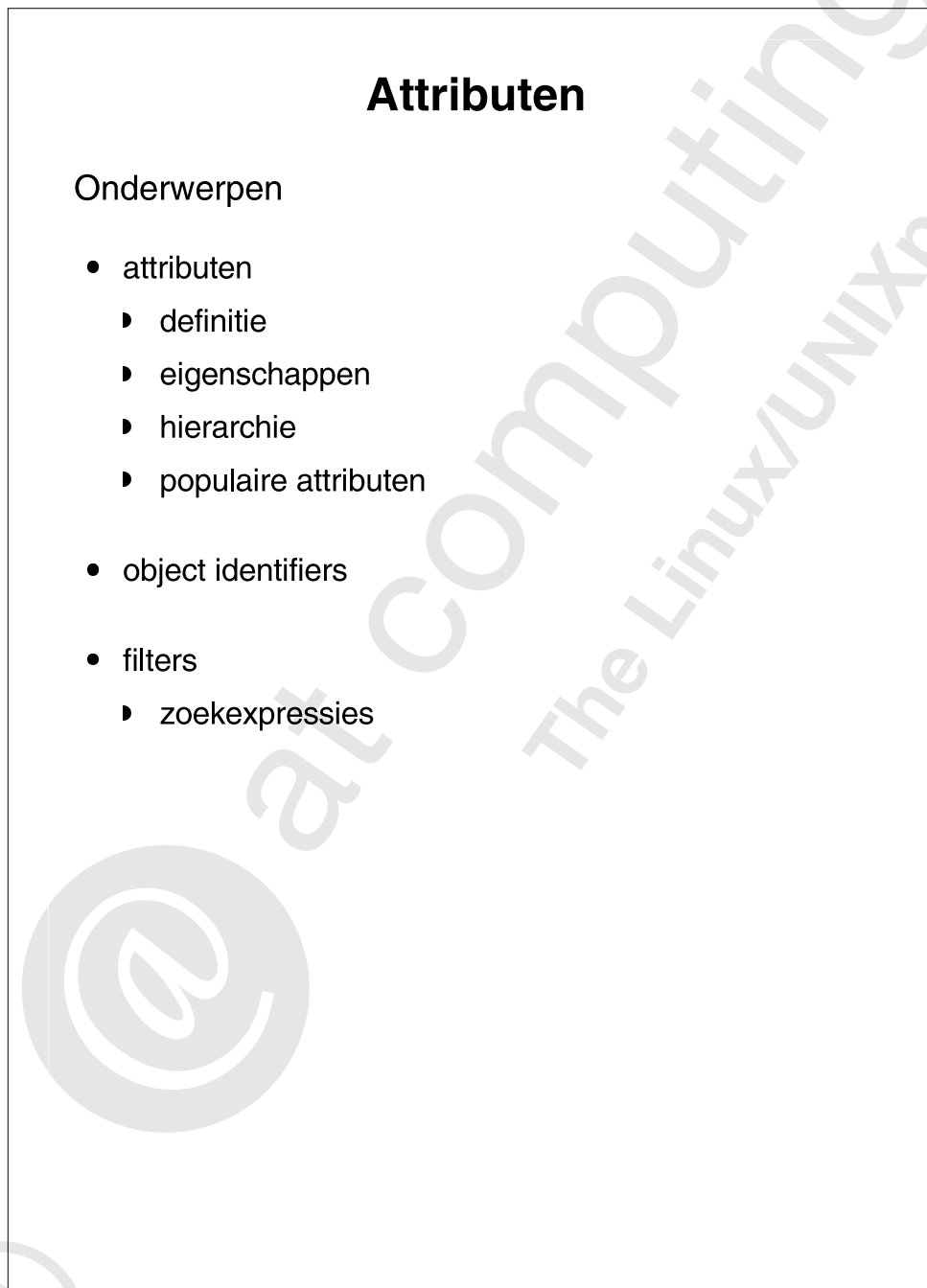
Nijmegen



Copyright © AT Computing 2005
Versie: 1b

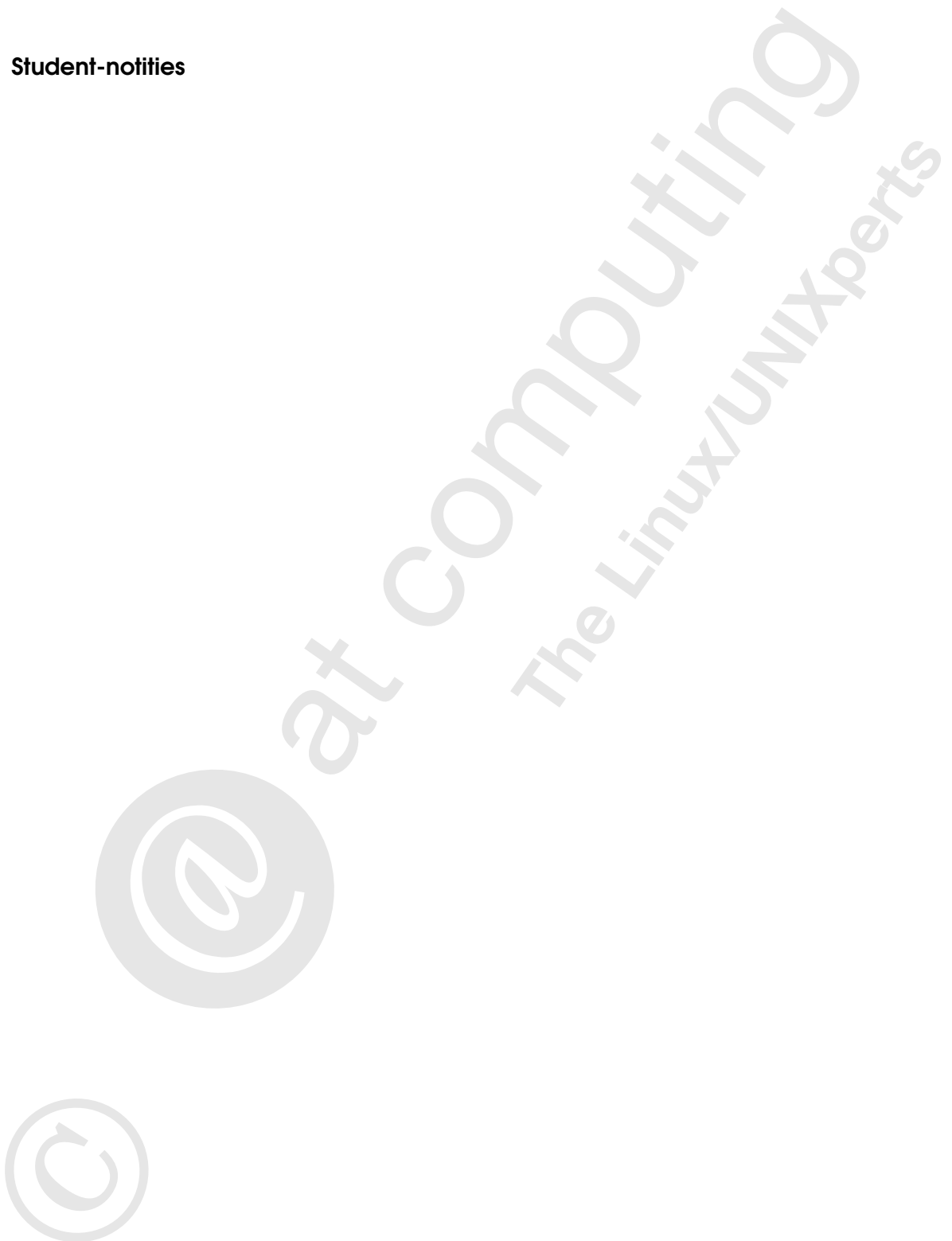
Student-notities





Figuur 1.

Student-notities



Attributen

Een entry is een verzameling attributen

- voorbeeld van een entry:

```
cn: Jan Klaassen
cn: Jan de Trompetter
mail: Jan.Klaassen@parcompu.nl
mail: jank@parcompu.nl
uid: jank
height: 180
gn: Jan
sn: Klaassen
```

- een attribuut heeft een naam
- eenzelfde attribuut mag meer dan eens voorkomen in dezelfde entry
- mogelijke waarden van attribuut zijn vastgelegd
- waarden van attributen kunnen vergeleken worden:
 - ▶ gelijkheid
 - ▶ ordening: kleiner/groter
 - ▶ substring
- eigenschappen van attributen zijn vastgelegd in definitie

Figuur 2.

Student-notities

Het attribuut `modifyTimestamp` is gedefinieerd in [RFC2252]. Dit attribuut is bedoeld om door de LDAP-server geschreven te worden, om zo van een entry bij te houden wanneer het voor het laatst gewijzigd is. Vandaar de specificatie dat het `SINGLE-VALUE` is, en `NO-USER-MODIFICATION`. De toevoeging `USAGE` geeft aan dat het een attribuut is dat algemeen is voor elke LDAP-server, en niet voor één specifieke.

`modifyTimestamp` is overigens een van de weinige attributen waarop een `ORDERING` gedefinieerd is — de meeste attributen kunnen niet met *groter* of *kleiner* vergeleken worden.

De attributen `name` en `commonName` zijn gedefinieerd in [RFC2256]. De specificatie `SUP name` bij `commonName` geeft aan dat het afgeleid is van `name` en derhalve alle eigenschappen van `name` erft, voorzover de eigen definitie die niet verandert.

Een volledig overzicht van alle standaard attributen vindt u op

<http://www.iana.org/assignments/ldap-parameters>

Voorbeeld Attributen

Definities uit core schema:

```
attributetype ( 2.5.18.2
  NAME 'modifyTimestamp'
  EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE
  NO-USER-MODIFICATION
  USAGE directoryOperation
)

attributetype ( 2.5.4.41
  NAME 'name'
  EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768}
)

attributetype ( 2.5.4.3
  NAME ( 'cn' 'commonName' )
  SUP name
)
```

- **modifyTimestamp** wordt door server gezet
- **name** is case-insensitive
- **commonName** is afgeleid van **name**
- **cn** is een alias voor **commonName**

Figuur 3.

Student-notities

Enige verduidelijking bij de syntax-beschrijving:

- De officiële definitie in [RFC2252] bevat tussen alle tekens ook nog optionele whitespace. Voor de overzichtelijkheid hebben we die hier weggelaten.
- Een *numericoid* is een nummerreeks, gescheiden door punten. Zie de sheet over object identifiers
- Een *oid* is een object identifier.
- NAME wordt gevolgd door een naam tussen enkele aanhalingstekens, of een lijst van zulke namen tussen haakjes.
- DESC wordt gevolgd door een beschrijving tussen enkele aanhalingstekens.
- De *noidlen* achter SYNTAX bestaat uit een *numericoid*, eventueel gevolgd door een maximum-lengte aanduiding.

De zogenaamde *matching rules* voor EQUALITY, SUBSTR en ORDERING hebben ook hun definities; deze vindt u echter niet in het schema van attribuut- en object-definities bij uw LDAP-server.

De eigenschap COLLECTIVE is een overblijfsel uit X.500 en wordt in LDAP niet gebruikt.

De eigenschap NO-USER-MODIFICATION geeft aan dat het attribuut niet door een gebruiker, maar alleen door de LDAP-server veranderd mag worden. Hiermee is het mogelijk om middels attributen bij te houden wie een entry heeft gemaakt of gewijzigd (attributen `creatorsName` en `modifiersName`) en wanneer het voor het laatst gewijzigd is (attribuut `modifyTimestamp`).

Syntax Attribuut Definitie

Definitie uit RFC2252:

```
AttributeTypeDescription = "("  
  numericoid                ; object-identifiek  
  [ "NAME" qdescrs ]       ; korte namen  
  [ "DESC" qdstring ]      ; beschrijving  
  [ "OBSOLETE" ]           ; niet actief  
  [ "SUP" oid ]             ; subtype  
  [ "EQUALITY" oid ]       ; gelijkheidsregel  
  [ "ORDERING" oid ]       ; volgorde-regel  
  [ "SUBSTR" oid ]         ; substring-regel  
  [ "SYNTAX" noidlen ]     ; syntax van waarde  
  [ "SINGLE-VALUE" ]       ; enkele waarde  
  [ "COLLECTIVE" ]        ; gezamenlijk  
  [ "NO-USER-MODIFICATION" ] ; niet wijzigbaar  
  [ "USAGE" AttributeUsage ]  
  extensions                ; uitbreidingen  
  ")"  
  
usage =  
  "userApplication" /  
  "directoryOperation" /  
  "distributedOperation" / ; DSA gezamenlijk  
  "dsaOperation"          ; DSA specifiek
```

- **SUP**: attribuut is afgeleid van ander attribuut
 - erft alle eigenschappen
 - eigenschappen mogen veranderd worden

Figuur 4.

Student-notities

Verreweg de meeste van deze attributen zijn afgeleid van name. De meeste zijn gedefinieerd in [RFC2256].

domainComponent is gedefinieerd in [RFC2247].

uid is gedefinieerd in [RFC2253].

uidNumber is gedefinieerd in [RFC2307] als deel van het zgn. NIS-schema, samen met allerlei andere attributen om UNIX gebruikersgegevens te modelleren. Uiteraard is uidNumber afgeleid van INTEGER.



Populaire Attributen

En hun korte namen

- **commonName, cn**
- **surname, sn**
- **givenName, gn**
- **countryName, c**
- **localityName, l**
- **stateOrProvinceName, st**
- **organizationName, o**
- **organizationalUnitName, ou**
- **domainComponent, dc**
- **userid, uid** : username, niet het nummer!
- **uidNumber**
- **userPassword**

Figuur 5.

Student-notities

De Object Identifiers is een erfenis van de CCITT (die nu ITU heet). De notatie van de nummerreeksen — nummers gescheiden met punten — is bedacht door de internet-gemeenschap. Deze identifiers vindt u ook terug bij SNMP, een ander protocol dat zijn oorsprong bij OSI-protocollen vindt.

Alle entiteiten in LDAP hebben zowel een naam (*oid*) als een nummerreeks: *numericoid*. De relatie bij attributen heeft u in de definitie van een attribuut gezien: elk attribuut heeft zowel een unieke naam als nummerreeks. Hetzelfde is het geval bij objecten. Bij de verschillende *matching rules* is dit ook het geval. Deze regels staan echter niet in het schema, maar zijn ingebakken in de server. In [RFC2252] vindt u echter een opsomming van de matching rules die tenminste herkend moeten worden.

Voor iedere prefix is er een autoriteit die langere numericoids mag bedenken naar eigen goeddunken. De numericoid 1.3.6.1 is in het algemeen in gebruik voor internet-protocollen; met name 1.3.6.1.1 (*directory*) is in gebruik voor LDAP-uitbreidingen.

De numericoid 1.3.6.1.4.1 is in gebruik voor bedrijven die eigen uitbreidingen willen bedenken. De IANA beheert deze numericoid en geeft nieuwe nummers uit aan wie er een wil. Zie

<http://www.iana.org/assignments/enterprise-numbers>

voor een overzicht. Bijvoorbeeld: 1.3.6.1.4.1.4203 is toegewezen aan de OpenLDAP organisatie. Verder vindt u bij

<http://www.alvestrand.no/objectid/>

een (incompleet) overzicht van toegewezen numericoids.

Numericoids worden bij LDAP ook nog gebruikt voor allerlei uitbreidingen op het LDAP-protocol. Een overzicht daarvan vindt u bij

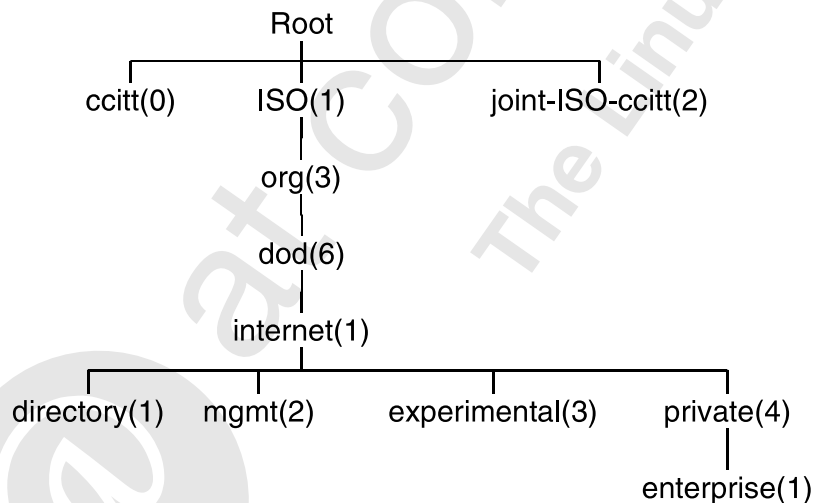
<http://www.iana.org/assignments/ldap-parameters>



Object Identifiers

Alle entiteiten in LDAP hebben een object identifier

- naam en een nummer: *numericoid*
- schema bedacht door CCITT (ITU) en ISO
- boomstructuur
 - 1.3.6.1: gebruikt door IETF
 - 1.3.6.1.4.1: geadministreerd door IANA



- bij LDAP gebruikt voor:
 - attributen, syntax van attributen
 - matching rules: EQUALITY, SUBSTR, ORDERING
 - objecten

Figuur 6.

Student-notities

Het commando `ldapsearch` kent veel opties die met authenticatie te maken hebben. De hier opgevoerde optie `-x` heeft daar ook mee te maken. Voor een overzicht van deze opties, zie hoofdstuk 5: *Praktijk: OpenLDAP*, waar de overige OpenLDAP-tools besproken worden.

Zoals op de sheet gebruikt, krijgt u bij `ldapsearch` alle attributen van een entry te zien die de gebruiker ook zou mogen wijzigen. U kunt echter `ldapsearch` ook een lijst van attributen meegeven die u wilt zien, bijvoorbeeld

```
ldapsearch '(gn=Jan)' gn,sn,height
```

In dat geval krijgt u alleen de gevraagde attributen. Er zijn ook twee wildcards mogelijk:

```
ldapsearch '(gn=Jan)' '*'
```

laat u alle door de gebruiker wijzigbare attributen zien;¹ en

```
ldapsearch '(gn=Jan)' +
```

laat u alle attributen zien die door de LDAP-server toegevoegd worden, zoals `modifiersName` en `modifyTimestamp`.

Als u een speciaal teken wilt gebruiken in de waarde van een attribuut, dan moet u daarvoor backslashes gebruiken; bijvoorbeeld

```
ldapsearch '(gn=Jan\2a)'
```

zoekt naar entries met attribuut `gn` met waarde `Jan*`. De string `2a` is de hexadecimale schrijfwijze voor de ASCII-waarde van `*`. Andere nuttige waarden: `\20` voor spatie, `\28` voor `(`, `\29` voor `)`, en `\5c` voor `\`.



1. Let op dat op de UNIX-commandoregel de asterisk `*` een bijzonder teken is!

Filters

Selecteer entries op basis van waarden attributen

- is een attribuut aanwezig?

```
$ ldapsearch -x '(mail=*)'
```

- heeft een attribuut een bepaalde waarde?

- attribuut moet **EQUALITY** hebben

```
$ ldapsearch -x '(gn=Jan)'
```

- attribuut-waarde bevat een bepaalde substring

- attribuut moet **SUBSTR** hebben

```
$ ldapsearch -x '(cn=*Jan*)'
```

- attribuut-waarde is groter dan...

- attribuut moet **ORDERING** hebben

```
$ ldapsearch -x '(height>=160)'
```

- combinatie van filters

- logische AND, OR en NOT

```
$ ldapsearch -x '(&(mail=*)(height>=160))'  
$ ldapsearch -x '(|(cn=*jan*)(sn=*klaas*))'  
$ ldapsearch -x '!(mail=*parcompu*)'
```

Figuur 7.

Student-notities

Merk op dat voor ordening alleen \geq en \leq toegestaan zijn, $>$ en $<$ niet. Indien u zo'n relatie zou willen gebruiken, bijvoorbeeld voor een filter (`height>200`), dan moet u dit omschrijven tot

```
(!(height<=200))
```

Merk voorts op dat de *logische connectieven* $\&$ en $|$ door een lijst van filters gevolgd worden; u kunt dus in één keer de logische AND van drie termen opschrijven, bijvoorbeeld

```
(&(cn=*Jan*)(height>=160)(st=Gelderland))
```

De syntax-regel voor *extensible* filters hebben we weggelaten; zie hiervoor [RFC2254]. Deze laten 2 dingen toe:

- om ook te matchen op de DN. Normaliter wordt alleen naar de attributen gekeken. Een voorbeeld hiervan: (`sn:dn=Barney`)
- om te matchen met een andere matching rule dan is voorgeschreven in de definitie van het attribuut. Een voorbeeld hiervan: (`cn:1.2.3.4.5:=Fred Flintstone`)

Wat een LDAP-server moet doen bij een *approximate* match is in de RFC's niet voorgeschreven. Meestal wordt het *soundex* algoritme van Knuth² hiervoor gebruikt, een eenvoudig algoritme dat aardig werkt voor het Engels maar absoluut niet voor andere talen.

2. Zie Donald Knuth, *The Art of Computer Programming*, vol. 3

Filter Syntax

Definitie uit RFC2254:

```

filter      = "(" filtercomp ")"
filtercomp = and / or / not / item
and         = "&" filterlist
or          = "|" filterlist
not         = "!" filter
filterlist = 1*filter
item        = simple /
             present /
             substring /
             extensible

simple       = attr filtertype value
filtertype = equal / approx / greater / less
equal      = "="
approx     = "~="
greater    = ">="
less       = "<="
present    = attr "="
substring  = attr "=" [initial] any [final]
initial    = value
any        = "*" *(value "*")
final      = value

```

- **extensible**: om andere matching regels te gebruiken
- **approx**: "klinkt als"

Figuur 8.

Student-notifies

