OpenLDAP

Linux Systems Authentication

Dr. Giuliano Taffoni – IASFBO

Layout

- Introduction to LDAP
- Authentication based on LDAP
  - Linux on Linux
- LDAP over SSL
- Fault Tolerance: basic replication.

LDAP Overview

- LDAP is a ‘Lightweight Directory Access Protocol’ (RFC 4510)
- LDAP marries a lightweight DAP with the X.500 information model
- Uses an extensible hierarchical object data model

What is LDAP

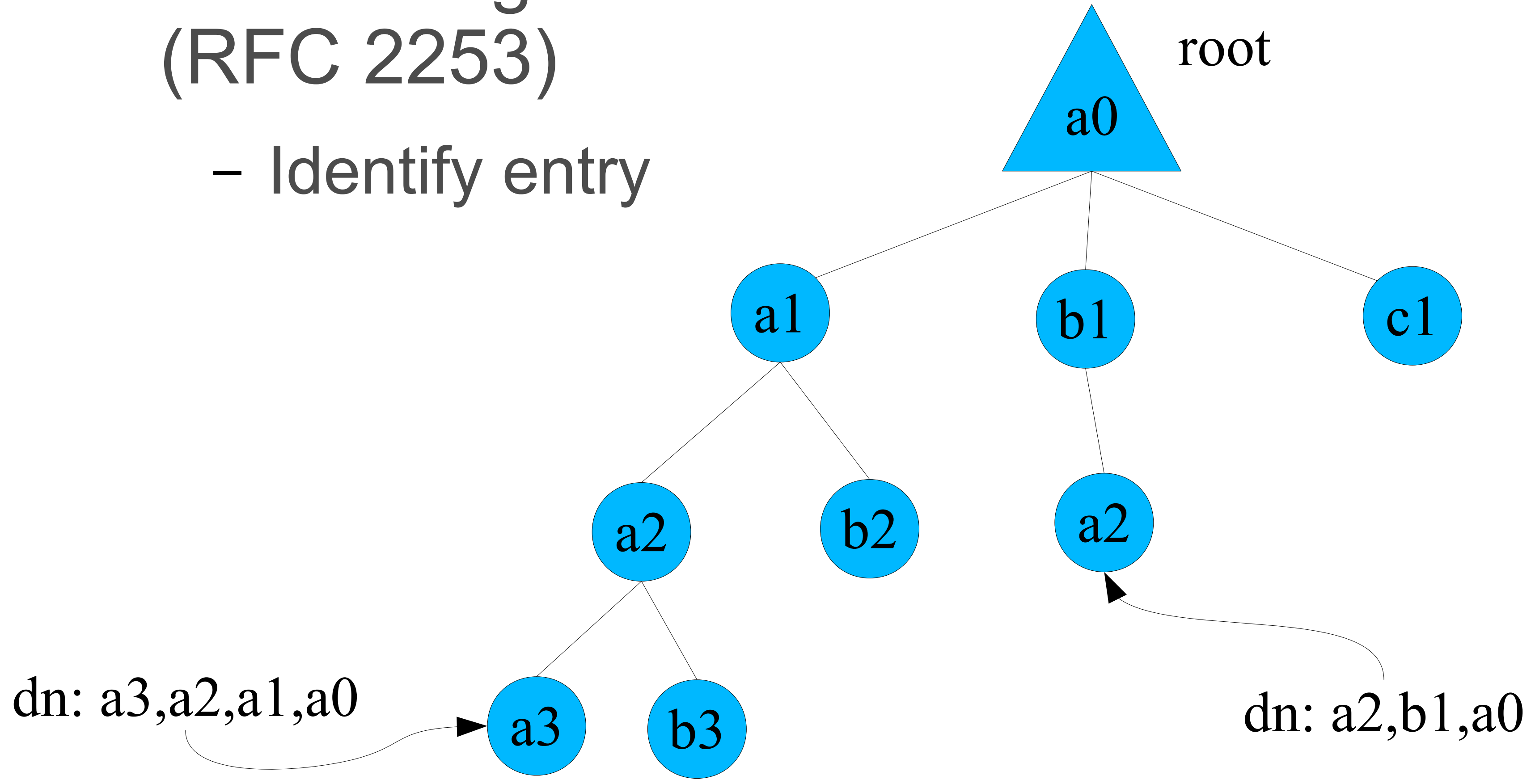
- LDAP is a directory service: information tree
- Front end to a DB.
- Implement multiple ‘back-ends’: **LDBM**, RDBMS, simple indexes (Berkeley DB), X.500 gateway
- Designed for frequent reads and infrequent writes

Authentication - LDAP

- username+passwd repository
- Store objects (ex. Jpegs, ssh certificates)
- Replication
- Load balancing (replicas)
- Not a db: no rollback, no transactions

Some terminology

- DN: distinguish name (RFC 2253)
  - Identify entry



Some terminology

- <attribute><value>
- Attribute has:
  - Name (unique, case insensitive)
  - OID (int = 1.3.6.1.4.1.1466.115.121.1.27)
- DN root identify the whole tree
- RDN (relative distinguish name)
  - Unique for the same progenitor
  - ex. “a2”
  - DN = sequence of RDNs

Naming

- Traditional naming:
  - c=it, l=bologna, o=iasf
- RFC naming
  - dc=it, dc=bologna, dc=inaf
  - dc=domain component

Schema and classes

- objectClass
  - Set of functional attributes
  - Each attribute is described by a class
  - Required attribute
  - Optional attribute

objectClass: Person

Required	**Optional**
cn	description
sn	seeAlso
	telephoneNumber
	userPassword

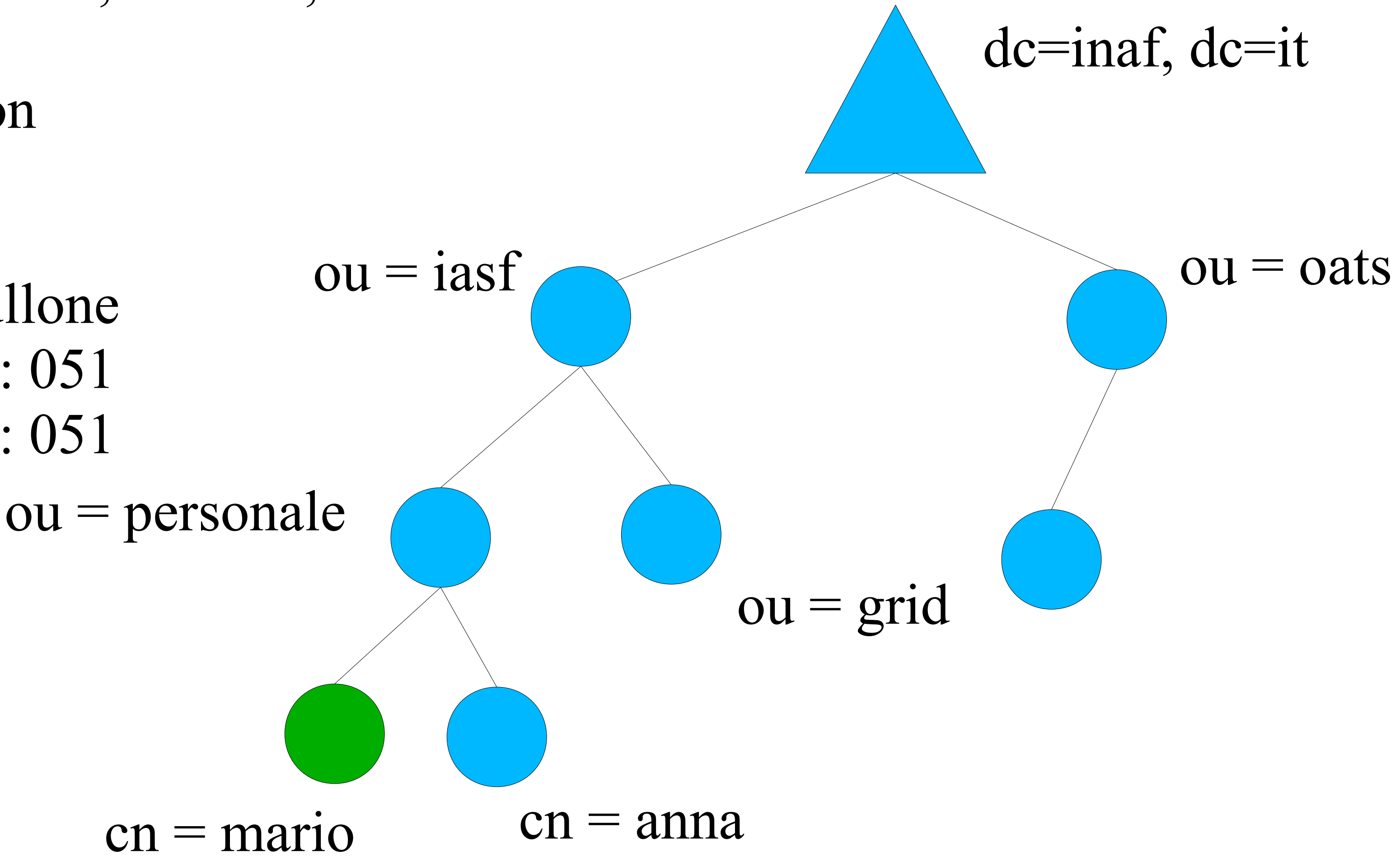
Schema and classes

organizationalUnit

Required	Optional
ou	businessCategory
	description
	telephoneNumber
	PostalAddress
	postalCode
	street
	etc....

Example

dn: cn=mario, ou=iasf, dc=inaf, dc=it
objectClass: top
objectClass: person
cn: mario
sn: rossi
description: fannullone
telephoneNumber: 051
telephoneNumber: 051



OpenLDAP

- Version 2.4.20
- Packages for Linux
  - openldap, openldap-servers, openldap-clients
  - 2.3.4 RHEL 5
  - 2.4 Ubuntu 9.10
  - 2.3.43 Gentoo 2009

OpenLDAP protocol

- client-server (message-oriented)
- client is able to make more than one request simultaneously (identified by a message ID)
- Messages are not exchanged in plain text but with a simplified version of BER (Basic Encoding Rules

OpenLDAP security

- ACL to support granularity access to the tree
- SSL mode
- TLS mode (StartTLS)

Note: If your server has SSL options loaded, this will launch a StartTLS capable daemon on port 389 (which is also capable of unencrypted communication) and a SSL only daemon on port 636.

Under StartTLS you are leaving the security of the system to the clients because the ldap:// is capable of unencrypted communication.

OpenLDAP

- Client cli
  - ldaptools:
    - ldapsearch, ldapadd, ldapmodify
    - Slapadd, slapindex, slappasswd
  - Scripts (/usr/share/openldap)
- Server
  - Slapd (stand-alone LDAP daemon)
  - Slurpd (replication Daemon)
- Configuration
  - /etc/openldap

Configuration

- /etc/openldap/schema - for schemas
- /etc/openldap/slapd.conf – for slapd
- /etc/openldap/ldap.conf - for clients that uses the system
  - /etc/ldap.conf

Schema

- File that collects classes
- Examples:
  - /etc/openldap/schema/core.schema
    - Person, orgUnit, etc.
  - /etc/openldap/schema/samba.schema
    - Samba attributes
  - /etc/openldap/schema/nis.schema
    - homeDirectory, loginShell

slapd.conf

- First include schemas

```
include        /etc/openldap/schema/core.schema
include        /etc/openldap/schema/cosine.schema
include        /etc/openldap/schema/inetorgperson.schema
include        /etc/openldap/schema/nis.schema
include        /etc/openldap/schema/samba.schema
include       /etc/openldap/schema/autofs.schema
```

slapd.conf

- backend definitions

```
# Load dynamic backend modules:
modulepath      /usr/lib/openldap/openldap
moduleload      back_hdb.so
backend         hdb

database        hdb
suffix          "ou=oats,dc=inaf,dc=it"

rootdn          "cn=manager,ou=oats,dc=inaf,dc=it"
rootpw          {MD5}....

overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100
directory       /var/lib/openldap-data
# Indices to maintain
index   cn,sn,uid                       eq,approx,sub
index   objectClass,entryCSN,entryUUID  eq
```

This specifies the DIT to manage

The database directory MUST exist prior to running slapd AND should only be accessible by the slapd and slap tools. Mode 700 recommended.

back-ends

- back-ldbm (classic) obsolete
- back-dbd
  - Back-bdb is a back end that is optimized for the Berkeley DB and takes advantage of its page locking features to improve concurrency. Load times are substantially improved and database sizes are halved.
- back-hdb
  - Back-hdb is a back end that is based on back-bdb, but which organizes its data in a true hierarchical fashion. Because of this, back-hdb supports the subtree rename operation, allowing subtrees to be quickly and efficiently moved within the same database (a requirement of the LDAPv3 standard which most other Directory Services packages fail to provide). Another advantage of back-hdb is that its hierarchical design makes for higher write throughput. This is especially good for applications that frequently modify the LDAP database.

indexing

- “index”: index the tree to increase performance

  ```
  index {list_attr} [pres,eq,approx,sub,none]
  ```

- Recommend: `index objectClass eq`

pres	Present (index per attributes (which object contains which attr))
eq	Equality (value equals to the filter)
approx	Approximate (similar)
sub	Substring (value that contains the search)
none	No index

slapd.conf

- ACL

  ```
  access to * by * read
  ```

  - More details in slide #Slide 23
  - Rootdn: necessary in slapd because at first startup there is no entry in the tree, neither the root one (we need to specify it somewhere). Later we can move it it the tree.
- slappasswd -h {MD5} -s the_password

Suggestions

- rootdn is mandatory
- proxyuser (suggested)
  - Can be any user (ex. Morgan)
  - Assign reading root capacities from ACLs
  - Use it to query the tree (more secure)
  - **NOTE**: Use root to insert and modify

ldap.conf

- Client configuration (/etc/openldap)
- **NOTE**: /etc/ldap.conf (used by PAM_LDAP, NSS_LDAP

```
base    ou=oats,dc=inaf,dc=it
URI     ldap://127.0.0.1/
```

LDAP server IP

note: URI ldaps:// for TLS
BASE: maybe a rootdn subtree

Start the server

- service ldap start (slapd start)
- chkconfig ldap on
- Or other ways according to the distribution

Populate the tree

- Ldaptools (cli)
- PhpLDAPadmin (http://phpldapadmin.sourceforge.net/wiki/)
- Webmin (http://www.webmin.com/)
- ldapvi (http://www.lichteblau.com/ldapvi/)

LDIF

- Ldap data exchange format (ASCII)

```
<dn:><distinguish_name>
<objectClass><value>
<attribute_RDN><value>
<attribute><value>
```

```
dn: uid=morgan,ou=oats,dc=inaf,dc=it
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
uid: morgan
loginShell: /bin/bash
uidNumber: 2001
gidNumber: 10
sn: Taffoni
cn: Giuliano Taffoni
homeDirectory: /home/morgan
```

LDIF format

```
<dn:><distinguish_name>
<objectClass><value>
<attribute_RDN><value>
<attribute><value>

```

Empty line

```
<dn:><distinguish_name>
<objectClass><value>
<attribute_RDN><value>
<attribute><value>

```

Empty line

```
<dn:><distinguish_name>
<objectClass><value>
<attribute_RDN><value>
<attribute><value>
```

LDIF with Italian names

- Note that LDIF does not support accents (ex. sn: Cartellà)
  - Transform in to UTF-8

    ```
    “iconv -f iso-8859-1 -t utf-8 source.ldif
      > dest.ldif”
    ```

- `May add images:`
  - `jpegPhoto: < file://path/name.jpg`

Create the LDIF files

- Create multiple files to manage multiple info: DC, Users, OU, groups etc.
- ex. /etc/openldap/ldif/[inaf.ldif, oats.ldif]

```
dn: dc=inaf,dc=it
objectClass: top
objectClass: organization
objectClass: dcObject
o: inaf.it
dc: inaf
```

```
dc: ou=oats,dc=inaf,dc=it
objectClass: organizationalUnit
ou: oats
```

Add Info in the tree

```
# ldapadd -x -D "cn=manager,ou=oats,dc=inaf,dc=it" -W
-f /etc/openldap/root.ldif

Enter LDAP Password:
```

users

```
dn: uid=taffoni,ou=oats,dc=inaf,dc=it
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
uid: taffoni
loginShell: /bin/bash
gecos: Taffoni Giuliano
sn: taffoni
homeDirectory: /home/taffoni
mail: taffoni@oats.inaf.it
cn: Taffoni Giuliano
TelephoneNumber: +39040....
userPassword: {crypt}....
uidNumber: 2002
gidNumber: 2415
```

Add entries

```
# ldapadd -x -D "cn=manager,ou=oats,dc=inaf,dc=it" -W
-f /etc/openldap/users.ldif
Enter LDAP Password:
```

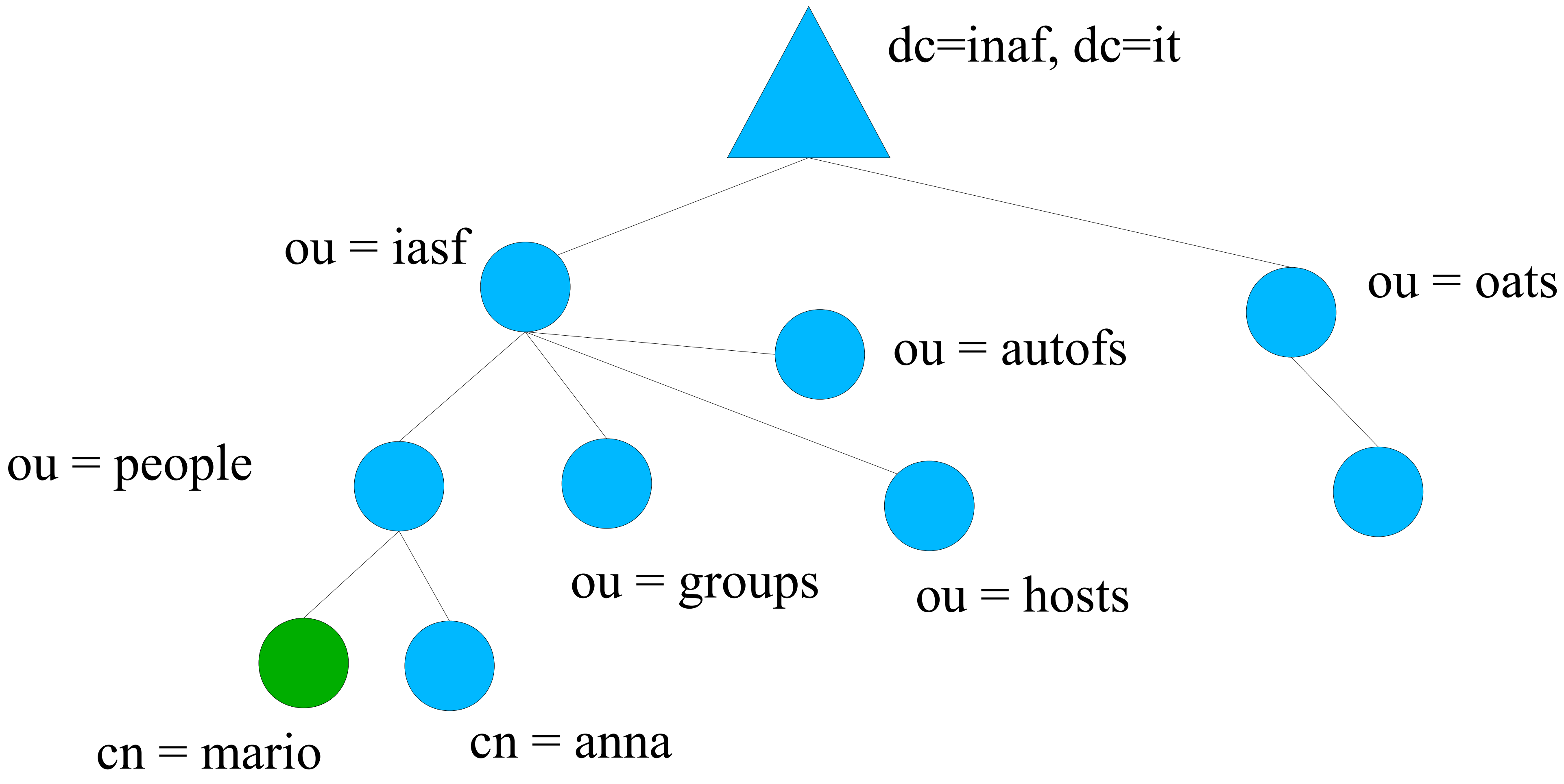
This takes a long time!!!!

ldapadd vs slapadd

- ldapadd
  - Connects to the frontend and modify back-end
  - On the fly
  - Uses ldif

- slapadd
  - Enters data into the back-end
  - **Stop the daemon!**
  - Uses ldif

Migrate from other systems



Migrate the entries

- Migration tools (www.padl.com)
- Perl scripts to migrate from local (YP) to LDAP
  - Install with openldap-server (commonly)
  - migrate-common.ph (conf file)
    - $DEFAULT_MAIL_DOMAIN="inaf.it"
    - $DEFAULT_BASE="ou=oats,dc=inaf,dc=it"
    - $EXTENDED_SCHEMA=1

tools

- migrate_base.pl
- migrate_passwd.pl
- migrate_group.pl
- migrate_hosts.pl

```
# ./migrate_base.pl > /etc/openldap/ldif/base.ldif
```

Edit the file to verify it.

Ldap in action: PAM and NSS

- pam_ldap + nss_ldap
  - /etc/pam.d/system-auth
  - /etc/nsswitch.conf
  - /etc/ldap.conf

system-auth

```
auth        required     pam_env.so
auth        sufficient   pam_ldap.so
auth        sufficient   pam_unix.so likeauth nullok nodelay \
use_first_pass
auth        required     pam_deny.so

account     sufficient   pam_ldap.so
account     required     pam_unix.so

password    required     pam_cracklib.so difok=2 minlen=8 \
dcredit=2 ocredit=2 retry=3
password    sufficient   pam_unix.so nullok md5 shadow use_authtok
password    sufficient   pam_ldap.so use_authtok
password    required     pam_deny.so

session     required     pam_limits.so
session     required     pam_unix.so
session     optional     pam_ldap.so
```

ldap.conf

```
ldap_version 3
scope sub
timelimit 3
bind_timelimit 3
bind_policy hard
idle_timelimit 3600
pam_login_attribute uid
pam_member_attribute gid
pam_password md5 pam_filter
uri ldap://server.hostname
suffix "dc=inaf,dc=it"
base ou=oats,dc=inaf,dc=it?sub
nss_base_passwd ou=oats,dc=inaf,dc=it?sub
nss_base_shadow ou=oats,ou=ts,dc=si.inaf,dc=it?sub
nss_base_group ou=groups,ou=oats, dc=inaf,dc=it?one?description=client.hostname
binddn cn=proxyuser,dc=inaf,dc=it
Bindpw .....
```

nsswitch.conf

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

Name service cache restart

```
# service nscd restart

# nscd --invalidate=TABLE
```

to invalidate cache

Test the configuration

- From client

```
# getent passwd
```

Increasing performance

- USUALLY IT IS NOT A PROBLEM
  - 300 users on one server (2 CPUs 3GHz Xeon, 4 GB RAM)
  - 40 users on one server (2 CPUS PIII 2 GB ram)
- But just in case....
  - tune your HW
  - works on indexes and DB
  - thinks about replication

the role of HW

- DUAL CPU (note that the threads directive in slapd may be increased for more cpus. default 16)
- Memory (direct impact in performances)
  - note that it is true for big db
  - cache should be tuned on the size of memory
- disk access speed
  - use different disks for db and logging

indexing

How does it works? If you're searching on a filter that has been indexed, then the search reads the index and pulls exactly the entries that are referenced by the index. If the filter term has not been indexed, then the search must read every single entry in the target scope and test to see if each entry matches the filter.

- ok: `index cn,sn,givenname,mail eq`
- “userPassword” is useless
- Presence may be dangerous!!!!!

If your client application uses presence filters and if the target attribute exists on the majority of entries in your target scope, then all of those entries are going to be read anyway. The presence index does absolutely **NOTHING** to benefit the search, just waists CPU and Memory.

Loglevel

- use the appropriate loglevel
  - loglevel 256 (default) is ok
  - loglevel 0 increase performances but not suggested
- logging may be useful:

```
"<= bdb_equality_candidates: (pippo) index_param
failed (18)"
```

  - application are using an equality filter pippo= something. add “pippo” index

memory and cache

- BDB cache size (A) necessary to load the database via slapadd in optimal time

```
du -c -h *.bdb (db size)
```

  -

- BDB cache size (B) necessary to have a high performing running slapd once the data is loaded
  - *id2entry.bdb* file, plus about 10% for growth
- IDL cache which is used for Index Data Lookups
- Importance issues: what should I fit into memory?
  - cache A + cache B + IDL
  - cache A + cache B
  - entry cache

calculate memory

- BDB uses 2 files: dn2id.bdb (8KBxpage), id2entry.bdb (16KBxpage)
- B-tree DB: balanced tree
- need enough cache to store all the internal nodes of the db
  - db_stat -d (# of pages)
- cache = # pages + some more for internal leaf data pages (given from db_stat)
- set_cachesize in DB_CONFIG file

slapd.conf

- cachesize = number of entries in memory
  - use a number that is comparable with the entries in your tree
- idlcachesize = cachesize

Adding some security and customization

Restrict access to users

- Problem:
  - I have one auth server and many clients. I do not want each user to connect to each client.
- Solution: restrict access thanks to LDAP capabilities

Create a custom schema

```
attributetype ( 1.3.6.1.4.1.22242.1.1.5
        NAME 'hostAccess'
        DESC 'Access Level'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

attributetype ( 1.3.6.1.4.1.22242.1.1.6
        NAME 'gridHomeDirectory'
        DESC 'cluster Homedir'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

objectclass   ( 1.3.6.1.4.1.22242.1.2.1
    NAME 'inaf'
        DESC 'INAF  LDAP schema'
        AUXILIARY
        MAY ( hostAccess $ gridHomeDirectory )
        )
```

string

But you may add your own customization. Verify the OID of the class!!!!

Example: wifiAccess

Client ldap.conf

```
ldap_version 3
scope sub
timelimit 3
bind_timelimit 3
bind_policy hard
idle_timelimit 3600
pam_login_attribute uid
pam_member_attribute gid
pam_password md5 pam_filter
uri ldap://server.hostname
suffix "dc=inaf,dc=it"
base ou=oats,dc=inaf,dc=it?sub
nss_base_passwd ou=oats,dc=inaf,dc=it?sub?hostAccess=client.hostname
nss_base_shadow ou=oats,ou=ts,dc=si.inaf,dc=it?sub?hostAccess=client.hostname
nss_base_group ou=groups,ou=oats, dc=inaf,dc=it?one?description=client.hostname
binddn cn=proxyuser,dc=inaf,dc=it
Bindpw .....
```

Users ldif

```
dn: uid=taffoni,ou=oats,dc=inaf,dc=it
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: inafsi
uid: taffoni
loginShell: /bin/bash
gecos: Taffoni Giuliano
sn: taffoni
homeDirectory: /home/taffoni
mail: taffoni@oats.inaf.it
cn: Taffoni Giuliano
TelephoneNumber: +39040....
userPassword: {crypt}....
uidNumber: 2002
GidNumber: 2415
hostAccess: client.hostname
hostAccess: client2.hostname
```

Automatic client configuration

Modify /etc/pam.d/sshd append at the end of the file this line:

```
session required pam_mkhomedir.so skel=/etc/skel umask=0077
```

This automatically creates the home at first login

Customize home

```
dn: uid=taffoni,ou=oats,dc=inaf,dc=it
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: inafsi
uid: taffoni
loginShell: /bin/bash
gecos: Taffoni Giuliano
sn: taffoni
homeDirectory: /home/taffoni
gridHomedirectory: /users/taffoni
mail: taffoni@oats.inaf.it
cn: Taffoni Giuliano
TelephoneNumber: +39040....
userPassword: {crypt}....
uidNumber: 2002
GidNumber: 2415
hostAccess: client.hostname
hostAccess: client2.hostname
```

Client ldap.conf

```
ldap_version 3
scope sub
timelimit 3
bind_timelimit 3
bind_policy hard
idle_timelimit 3600
pam_login_attribute uid
pam_member_attribute gid
pam_password md5 pam_filter
uri ldap://server.hostname
suffix "dc=inaf,dc=it"
base ou=oats,dc=inaf,dc=it?sub
nss_map_attribute    homeDirectory gridHomeDirectory
nss_base_passwd ou=oats,dc=inaf,dc=it?sub
nss_base_shadow ou=oats,ou=ts,dc=si.inaf,dc=it?sub
nss_base_group ou=groups,ou=oats, dc=inaf,dc=it?one?description=client.hostname
binddn cn=proxyuser,dc=inaf,dc=it
Bindpw .....
```

LDAP with TLS SSL

- Get the certificates:
  - Server cert, server key, ca cert
- GARR, INFN, myca (xca)
- Modify server slapd.conf

```
security tls=1

TLSCertificateFile          /etc/openldap/ssl/cert.pem
TLSCertificateKeyFile       /etc/openldap/ssl/req.pem
TLSCACertificateFile        /etc/openldap/ssl/cacert.pem
```

Client configuration

- Modify /etc/openldap/ldap.conf

```
HOST myserver.com
PORT 636

TLS_CACERT /etc/ssl/certs/cacert.pem
TLS_REQCERT demand
```

- Sometimes it is necessary to set .ldaprc for root (both in /root and in /)

More security

- Client-server SSL authentication
  - Both client and server needs a server cert/key
- On server (slapd.conf)

```
TLSVerifyClient demand
```

- On client (ldap.conf and .ldaprc)

```
TLS_REQCERT demand

# client authentication
TLS_CERT /etc/ssl/ldap/client.cert.pem
TLS_KEY /etc/ssl/ldap/client.key.pem
```

testing

```
# openssl s_client -connect localhost:636 -showcerts
-state -CAfile <ca cert>
```

If you use client-server SSL

```
# openssl s_client -connect myserver.com:636 -state \
-CAfile /etc/ssl/ldap/cacert.pem \
-cert /etc/ssl/ldap/client.cert.pem \
-key /etc/ssl/ldapclient.key.pem
```

Search the tree

```
# ldapsearch -x -b "ou=oats,dc=inaf,dc=it" -D
"uid=morgan,ou=infra,ou=oats,dc=inaf,dc=it" '(uid=morgan)'
-H ldaps://localhost -W
```

Searching the tree

ldapsearch [*optional_options*] [*optional_search_filter*] [*optional_list_of_attributes*]

```
# ldapsearch -x -b "dc=inaf,dc=it" -D
"uid=morgan,ou=infra,ou=oats,dc=inaf,dc=it" '(uid=morgan)' -H
ldaps://localhost -W uid userPassword
```

```
access to dn.subtree="dc=inaf,dc=it" attrs="userPassword"
  by dn.subtree="ou=infra,ou=oats,dc=inaf,dc=it"  write
```

Using filters

- Basic syntax

```
attribute operator value
```

  - Operators: <, >, ~=,=, *, =*

- Multiple filters (boolean operators)

(***Boolean-operator(filter)(filter)(filter)***...)

- examples:

```
description=*X.500*

(&(!(objectClass=person))(cn~=printer3b))
```

ACL in practice

- slapd.conf
- access to (what) by (who) access control

what

*	all	
attr	Specific attributes	
dn	dn	
dn.scope	base	itself
	children	Only childrens
	subtree	itself+childrens

who

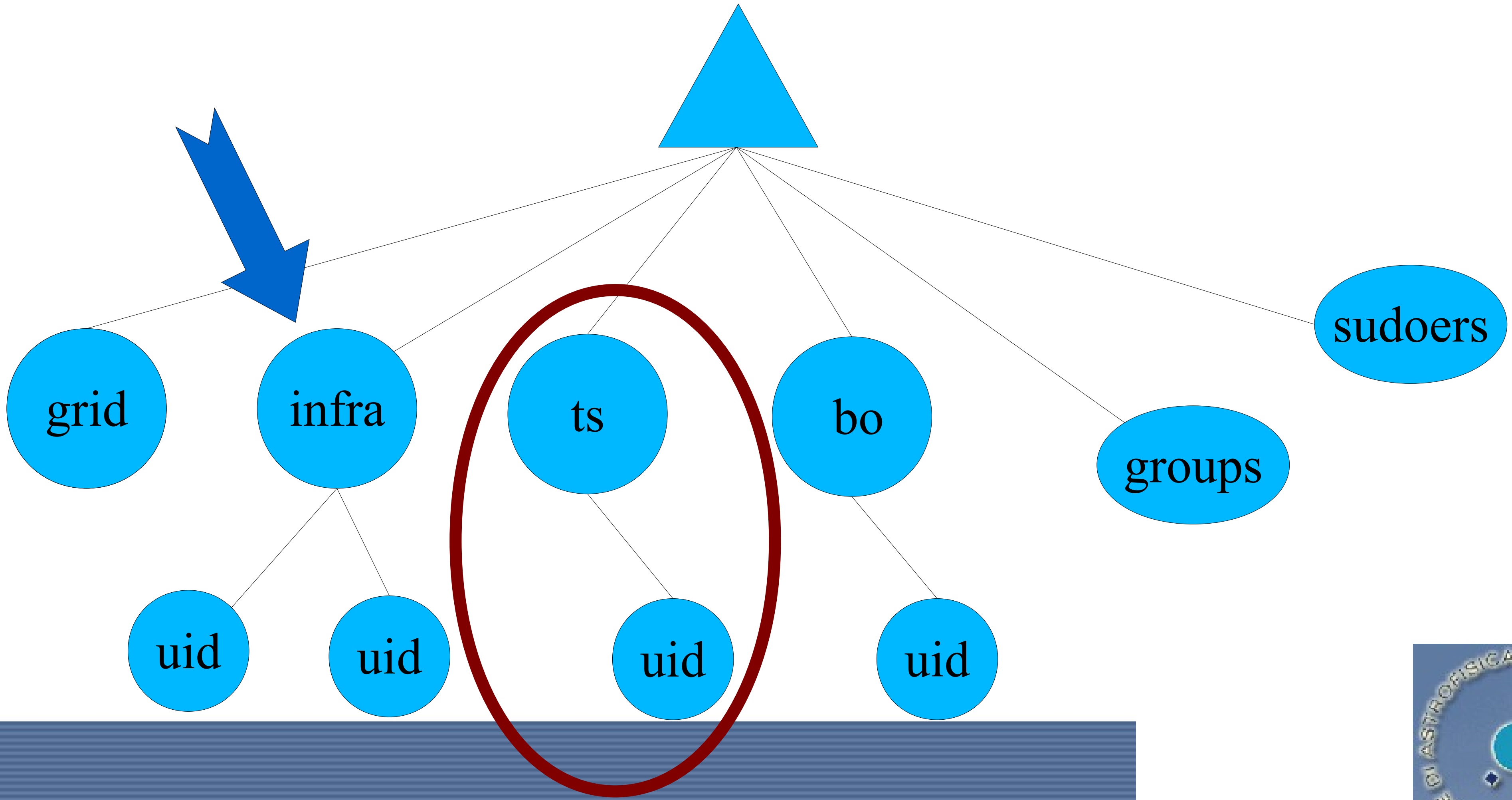
anonymous	anonymous
*	all
dn	A dn
dn.scope	See “what” table

control

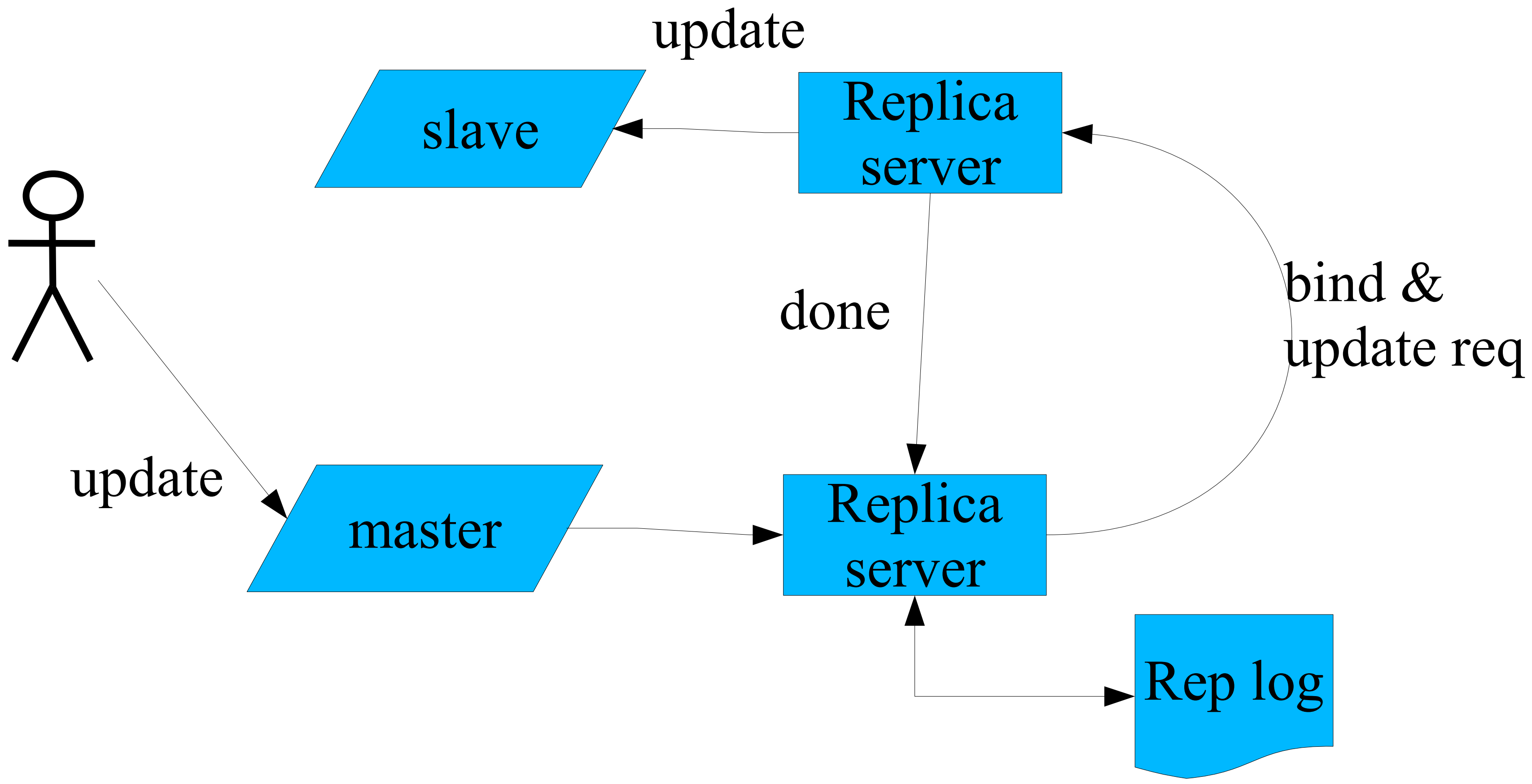
write	
read	
search	
compare	

example

access to dn.subtree="ou=ts,dc=inaf,dc=it" by dn.subtree="ou=infra,dc=inaf,dc=it" ... write



Basic replica



Replica master

- slapd.conf
  - replogfile “filename”
  - replica host = slave:389
  - binddn = “cn=Replicator, dc=inaf, dc=it”
  - SSL? “tsl = yes”
- NOTE: make a backup of the tree to a file (ldapxxx) and copy it to the slave

Replica slave

- Use the same slapd.conf of master
- But:
  - Delete the “replica host” statement
  - Delete the “replogfile” statement
  - updateref = masterIP
  - updatedn = “master binddn” (ex. cn=replicator....)
- Need ACL

Slave setup

- ACLs:

  access to dn=".*,dc=inaf,dc=it" by dn="cn=replicator, ..." write

- LDIF:
  - ldapadd the master ldif
  - But: add the replicator

```
dn: cn=replicator, dc=...
cn=replicator
objclass= top
objclass = simplesecurityobject
```

OpenLDAP components

- opendap cli:
  - ldapadd
  - Ldapsearch
- slap cli:
  - slapadd
- slapd: server daemon
- slurpd: replica daemon

Next issue

- Hyper-security on WAN: client-server double authentication
- Advanced replication
- Linux and Windows coexistence
  - SAMBA and LDAP
  - ACTIVE directory and Linux