What Is Identity Management

Identities

Definizione di *identity*

- Chiamiamo *identity* l'insieme di attributi che rappresentano una persona in un sistema informatico

Identities

- La definizione di una *identity* nell'intero sistema informatico o in uno specifico sistema equivale a una *proiezione* della persona nel sottospazio degli attributi rappresentabili nel sistema stesso

- Il sottinsieme di attributi che costituiscono la proiezione dell'identità in uno specifico sistema prende anche il nome di *identità immagine*

Identity Repositories

Definizione di Identity Repository

- Un sistema o sottosistema atto a contenere e rendere accessibili le identità immagine si chiama genericamente *identity repository*
- Fornisce strumenti, API e protocolli per creare e cercare le identità, modificarne gli attributi, costruire relazioni e definire regole per il controllo di queste operazioni

Identity Repositories

Esempio: Sistemi "*nix"

I file /etc/passwd ed /etc/group e le utility per la gestione degli utenti costituiscono l'*identity repository* del sistema

- getpwent(), setpwent()
- Network Information Service
- Pluggable Authentication Modules

Identity Repositories

Esempi: Sistemi Microsoft

In NT esistono il SAM database e le relative utility di amministrazione, sia della *security* locale che di dominio. Windows 200x è basato invece su Active Directory

- SMB
- ADSI
- LDAP

Identity Repositories

Esempi: Sistemi Novell

Novell Directory Service è stata una delle prime directory disponibili per ambienti di rete PC

- X.500
- NDS
- LDAP

Identity Repositories

Esempio: Sistemi IBM

Nei sistemi *mainframe* esiste RACF (Resource Access Control Facility), che controlla l'accesso a tutti i dati e i componenti del sistema con politiche di Mandatory Access Control

- LDAP

Directory

Molti sistemi possono contenere *digital identity* (es.: *database*) e renderle in qualche modo accessibili

Chiamiamo *directory* i sistemi il cui accesso è definito da un protocollo di comunicazione DAP

- Directory Access Protocol

LDAP directory

LDAP: un protocollo o una directory?

Contiene oggetti che hanno un tipo, sono identificati da una chiave unica (*distinguished name*) e sono corredati di attributi.

I tipi di oggetti più comuni sono

User,

Group,

Organizational Unit,

Organization, Country, Domain Context…

LDAP directory

Alcuni oggetti hanno il ruolo di contenitori, come Domain Context (DC), la Organizational Unit (OU) e la Organization (O), gli altri sono oggetti semplici e il loro nome è un Common Name (CN).

La directory ha un'organizzazione ad albero, che viene costruito mediante gli oggetti “contenitori”. Il nome di questi oggetti compone il distinguished name, ovvero la chiave di accesso, di tutti gli oggetti della directory, ad esempio

CN=Paolo Rossi, OU=Vendite, O=Fiat

LDAP directory

Warning

E’ importante inoltre osservare che, al contrario per esempio delle Organizational Unit, gli oggetti Group non sono contenitori. Hanno invece la caratteristica utile di essere correlabili ad altri oggetti (per esempio User) mediante l’attributo “members”, duale dell’attributo “memberOf” degli oggetti correlati.

E’ compito della Directory mantenere consistenti questi legami nel tempo.

LDAP directory

La descrizione sintattica degli attributi che possono esistere e l'assegnazione degli attributi associabili ai diversi tipi di oggetti è contenuta nel *directory schema*

Esistono anche ACL che permettono di esplicitare regole per l'accesso agli attributi

LDAP directory

La *directory* è acceduta mediante alcune funzioni primitive:

- ldapsearch
- ldapadd
- ldapmodify
- ldapdelete

I dati vengono forniti al momento della chiamata in opportune strutture o file esterni in formato standard (LDIF)

LDAP directory

Warning

E’ necessario rendersi conto del fatto che

i distinguished name non sono attributi

ma solamente le chiavi di accesso degli oggetti.

Non si può dunque cercare un oggetto per distinguished name, si può solo referenziarlo

Authentication

Le persone accedono alle risorse sulla base della loro *identity*

Il processo di verifica della *identity*, basata sulla disponibilità di credenziali, si chiama *authentication*

- *user/password*
- *digital certificates*
- *one-time passwords*, *tokens*, biometrics...

Authorization

Il processo che controlla gli accessi alle risorse da parte di una *identity*, si chiama *authorization*

- *coarse-grained* (statica, di provisioning)
- *fine-grained* (dinamica, di business)

Authorization

Provisioning rule

In una *directory* possiamo rappresentare con efficacia la *coarse-grained authorization* (mettendo in relazione utenti e gruppi e creando ACL) per garantire che un utente possa usare un dato o una applicazione; è un'autorizzazione *statica*

Business rule

In una applicazione invece la *fine-grained authorization* si effettua richiamando un algoritmo, che permette di autorizzare quell'utente a modificare quel dato in quel momento; è un'autorizzazione *dinamica*

Authorization

Per chiarire la differenza fra *coarse-grained* e *fine-grained* authorization si pensi al ruolo di un magazziniere di produzione, cioè colui che preleva i pezzi dal magazzino in funzione delle necessità produttive del momento.

E' necessaria una *coarse grained authorization* che gli consenta di eseguire l'applicazione "Prelievi di magazzino" (tutti i magazzinieri devono essere autorizzati a usare questa applicazione).

L'applicazione stessa implementerà poi una *fine grained authorization* che agisce istantaneamente ad ogni prelievo e permette o meno al magazziniere di scaricare ciascun pezzo in quel particolare momento (in quanto non è detto che qualunque pezzo sia sempre prelevabile dal magazzino o possa essere effettuato da ogni magazziniere; pensare per esempio ai turni di servizio).

Authorization

Il legame tra questi tipi di autorizzazioni e le directory è particolarmente significativo, in quanto

Le autorizzazioni statiche, cioè di tipo *coarse-grained* possono essere rappresentate in una directory, mentre quelle dinamiche o *fine-grained* no

Provisioning

Insieme con le identity si rappresentano anche altri oggetti:

- gruppi e unità organizzative
- sistemi (computer, stampanti, ecc)
- applicazioni e dati
- altre risorse (certificati, ecc.)

Si rappresentano inoltre relazioni e dipendenze tra di essi

Provisioning

Si chiama *provisioning* l'obiettivo di garantire la disponibilità delle risorse che occorrono per svolgere un compito

All'utente autenticato viene cioè reso disponibile il sottinsieme di risorse giusto in quel particolare momento

Identity Management

Identities

- Chiamiamo *identity* l'insieme di attributi che rappresentano una persona in un sistema informatico

- La definizione di una *identity* nell'intero sistema informatico o in uno specifico sistema equivale a una *proiezione* della persona nel sottospazio degli attributi rappresentabili nel sistema stesso

- Il sottinsieme di attributi che costituiscono la proiezione dell'identità in uno specifico sistema prende anche il nome di *identità immagine*

Identity Repositories

Come già visto, le identità elettroniche sono contenute e gestite all'interno di Identity Repository

- Sono sistemi atti a contenere e rendere accessibili le identità immagine
- Possono inoltre fornire strumenti, API e protocolli per creare e cercare le identità, modificarne gli attributi e costruire relazioni

Identity Repositories

Esempio: Sistemi "*nix"

I file /etc/passwd ed /etc/group e le utility per la gestione degli utenti costituiscono l'*identity repository* del sistema

- getpwent(), setpwent()
- Network Information Service
- Pluggable Authentication Modules

Identity Repositories

Esempi: Sistemi Microsoft

In NT esistono il SAM database e le relative utility di amministrazione, sia della *security* locale che di dominio. Windows 200x è basato invece su Active Directory

- SMB
- ADSI
- LDAP

Identity Repositories

Esempi: Sistemi Novell

Novell Directory Service è stata una delle prime directory disponibili per ambienti di rete PC

- X.500
- NDS
- LDAP

Identity Repositories

Esempio: Sistemi IBM

Nei sistemi *mainframe* esiste RACF (Resource Access Control Facility), che controlla l'accesso a tutti i dati e i componenti del sistema con politiche di Mandatory Access Control

- LDAP

Identity Repositories

Esiste un'altra classe di Identity Repository che non possiamo trascurare nell'ottica della gestione delle identità; potremmo definirla "repository di fatto"

- Tabelle "employee" nei database, sistemi gestionali, paghe, applicazioni MS Access ecc.
- Tabelle utente/password di applicazioni, fogli elettronici protetti, ZIP files o equivalenti vari
- Elenchi telefonici interni, database dei centralini, indirizzari ecc.
- Altri elenchi e anagrafiche, anche in formato cartaceo

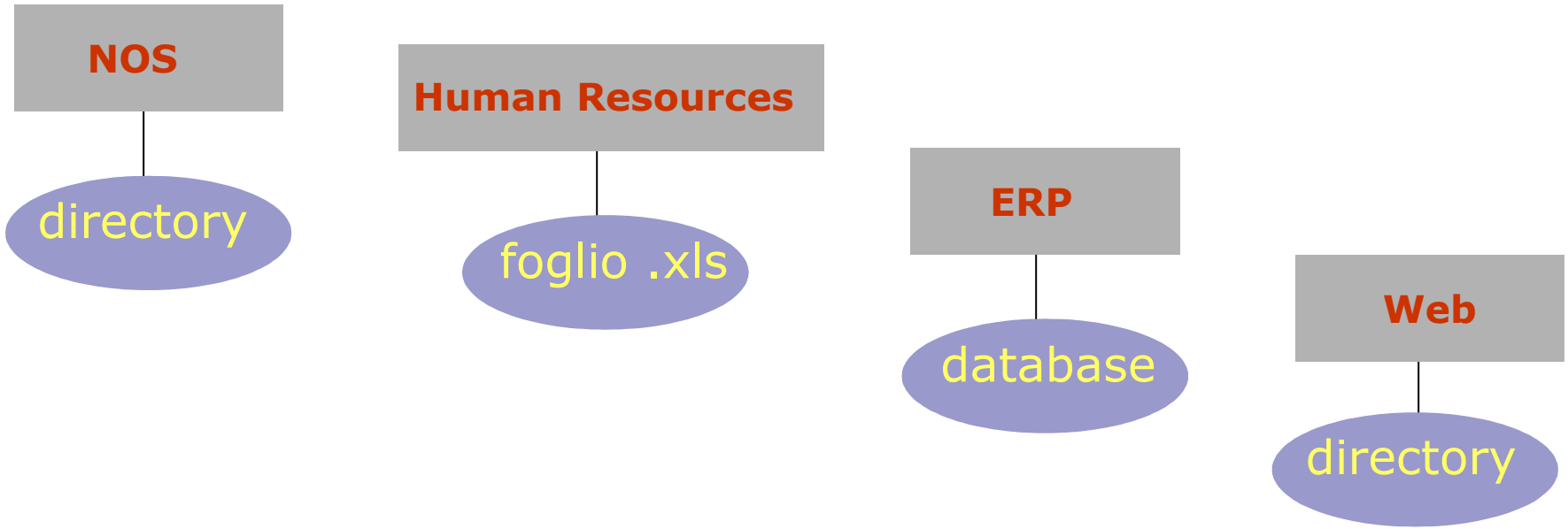
Riconoscere i *repository*

Non tutti gli esempi di repository di identità esposti precedentemente sono directory in senso proprio

Tuttavia tutti questi sottosistemi sono accessibili e manutenibili in vari modi (p.es. anche solo manualmente) e contenengono informazioni relative a utenti di sistemi o applicazioni, ossia contengono identità immagine di persone che interagiscono con il sistema informatico

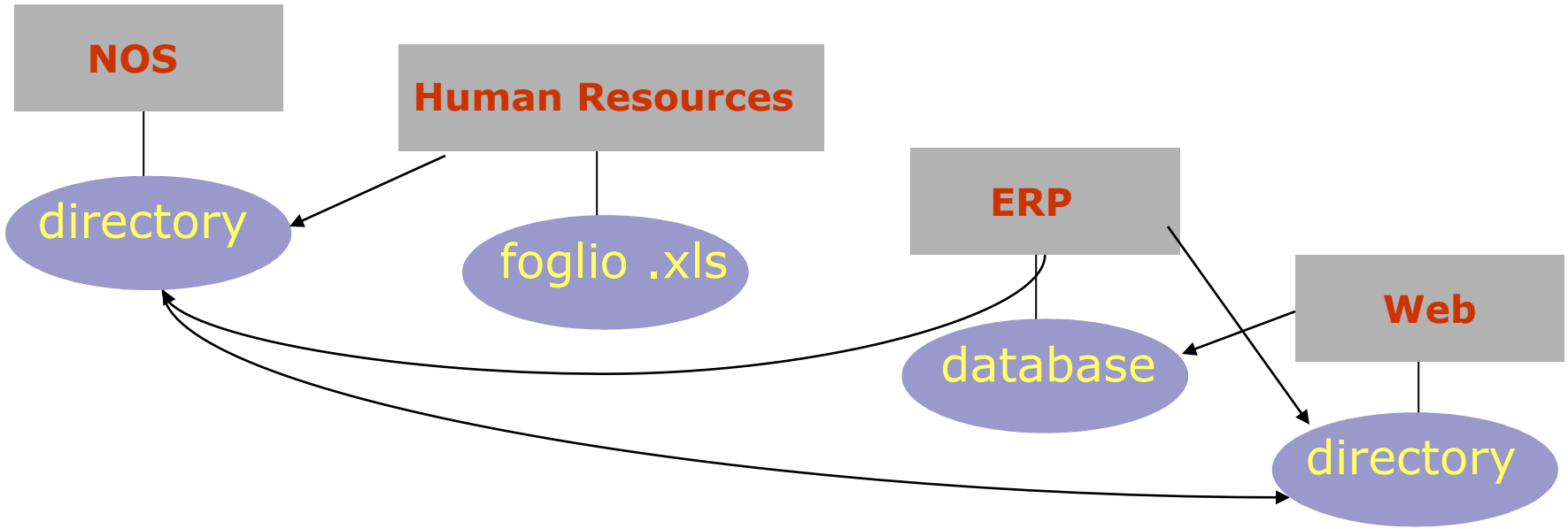
Riconoscere i *repository*

Tutti sono *identity repository*



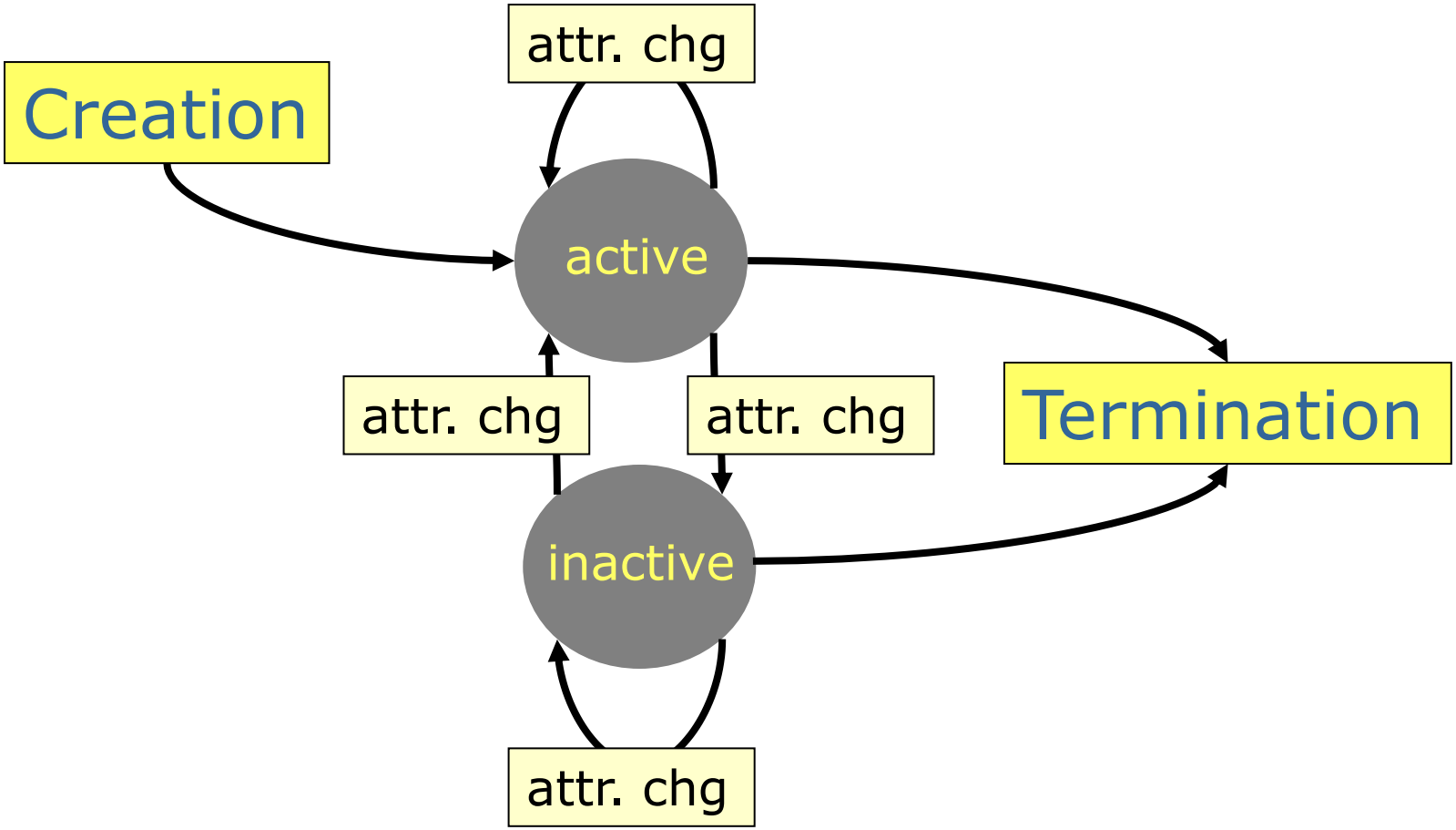
Riconoscere i *repository*

Ma sono gestiti in modo consistente?



???

Identity Lifecycle



Identity Lifecycle

Vari fattori concorrono a rendere onerosa la gestione consistente delle identity:

- numero dei sistemi
- diverse interfacce disponibili
- cambiamenti
- organizzazione delle relazioni
- diverse Entità partecipanti

Identity Lifecycle: requisiti di gestione

Per ogni identità immagine e per ogni suo attributo occorre garantire

Vincoli:

existence

ownership

context

Funzionalità:

connectivity

brokerage

… in tutti i sistemi e in ogni momento

Existence

Le identità non devono esistere necessariamente in tutti i sistemi, ma in quelli in cui l’utente reale è tenuto ad operare sì

Ownership

Gli attributi che definiscono le identità possono esistere in più sistemi, ma devono essere modificati dalle Entità giuste al momento giusto

Se necessario, le modifiche devono essere propagate nelle altre “immagini” dell'identità in modo consistente

Context

Le definizioni degli attributi delle identità immagine devono essere le stesse in tutti gli ambienti

Classico caso è invece mettere nella “descrizione” di un oggetto, che di solito è un campo libero è privo di significato, altre info, per esempio il numero di telefono. Questo porta rapidamente alla comparsa di informazioni inconsistenti nel sistema, in quanto dove l’informazione è gestita correttamente resterà aggiornata nel tempo, mentre il numero di telefono nella “descrizione” resterà sempre quello inserito inizialmente

Goal

Rispettare i vincoli e le funzionalità...

existence

ownership

context

connectivity

brokerage

Governare l'accesso all'informazione ...

authentication

provisioning

authorization

managed identity

Authentication

Le persone accedono alle risorse sulla base della loro *identity*

Il processo di verifica della *identity*, basata sulla disponibilità di credenziali, si chiama *authentication*

- *user/password*
- *digital certificates*
- *one-time passwords*, *tokens*, biometrics...

Provisioning

Si chiama *provisioning* l'obiettivo di garantire la disponibilità delle risorse che occorrono per svolgere un compito

All'utente autenticato viene cioè reso disponibile il sottinsieme di risorse giusto in quel particolare momento

Authorization

Il processo che controlla gli accessi alle risorse da parte di una *identity*, si chiama *authorization*

- *coarse-grained* (statica, di provisioning)
- *fine-grained* (dinamica, di business)

Identity Management

Sono stati fatti molti sforzi e vari tentativi per unificare gli *identity repository*. Si è però sempre osservato che:

- le applicazioni esistenti non si modificano facilmente
- è meglio mantenere le informazioni nei formati più appropriati
- vi sono vincoli politici e organizzativi che impediscono una completa unificazione

Esiste dunque il problema di mantenere le immagini di una *identity* in posti differenti e di fare in modo che diversi *identity repository* funzionino insieme mantenendo consistenti le informazioni sulle identità immagine che contengono

Identity Management

Connettività e brokering

- DAP, SQL, proprietary protocol + app + API
- propagare i cambiamenti e allineare i repository

Aggregazione e correlazione

- consolidare più repository in uno
- mantenere legami di correlazione
- garantire il rispetto della ownership

Gestione errori e replicazione

- integrità referenziale tra repository
- distribuire informazioni consistenti

Soluzioni

Meta-Directory

Multi-Directory Access

Synchronization Connectors

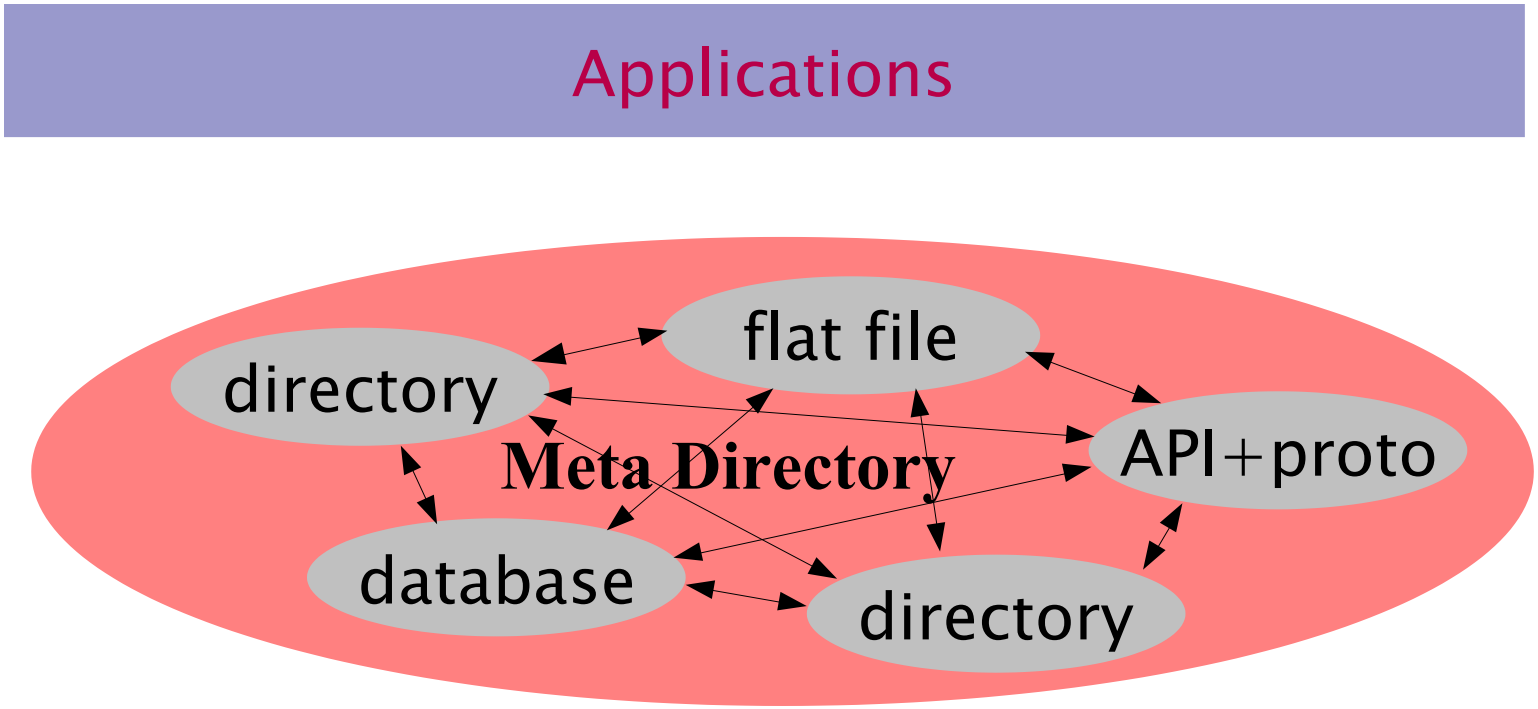
DAP Proxies

Directory Consolidation

Hub & Spokes

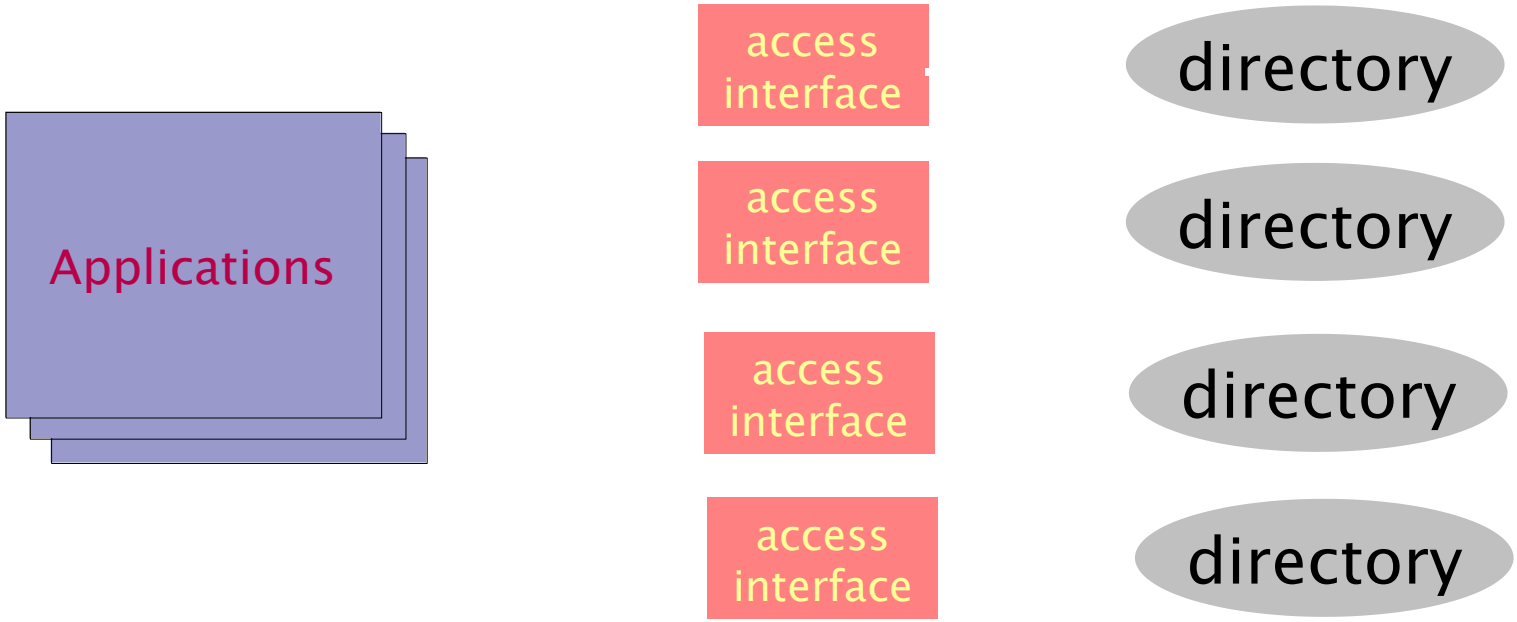
Meta Directory

- incapsula trasparentemente tutti i *repository*
- offre un'interfaccia opportuna alle applicazioni



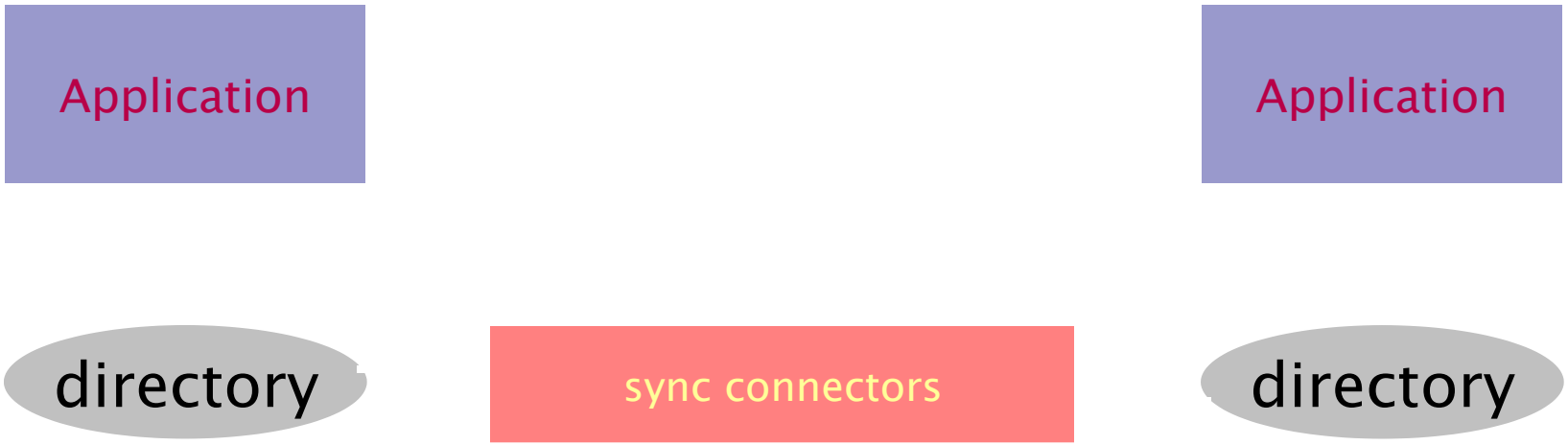
Multi-Directory access

- ciascuna applicazione interagisce direttamente con i *repository* contenenti le identità immagine ad essa pertinenti



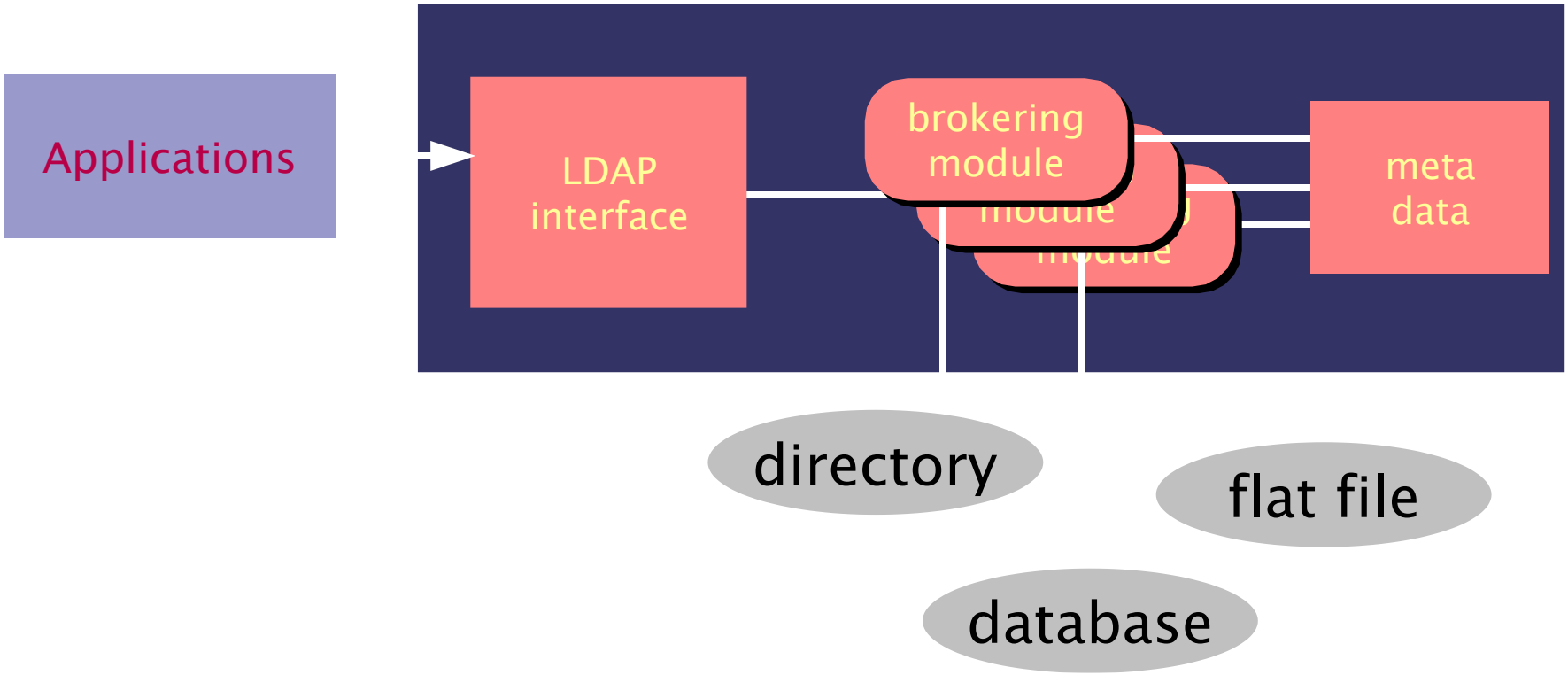
Synchronization Connectors

- le applicazioni accedono alle proprie *directory*
- le *directory* vengono sincronizzate dai connettori



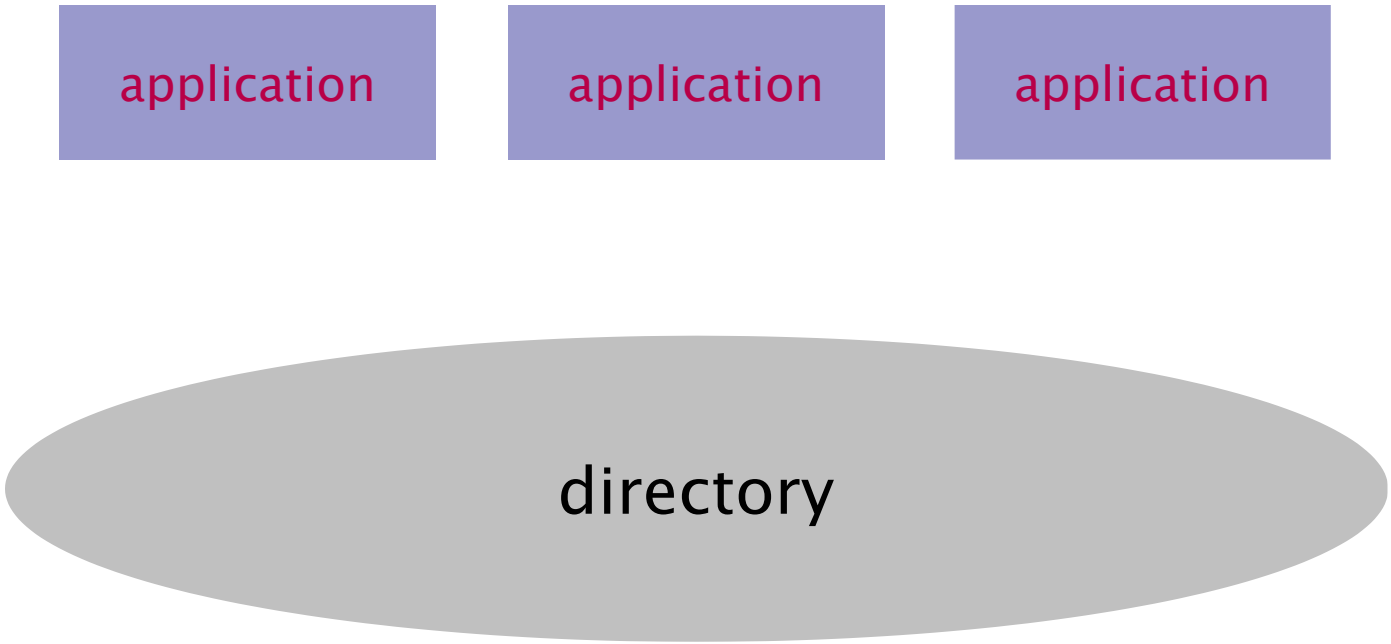
DAP Proxies

- accesso alle *directory* con un'interfaccia comune
- protocollo di accesso standard (es. LDAP), con *brokering module* interfacciati con le varie *directory*



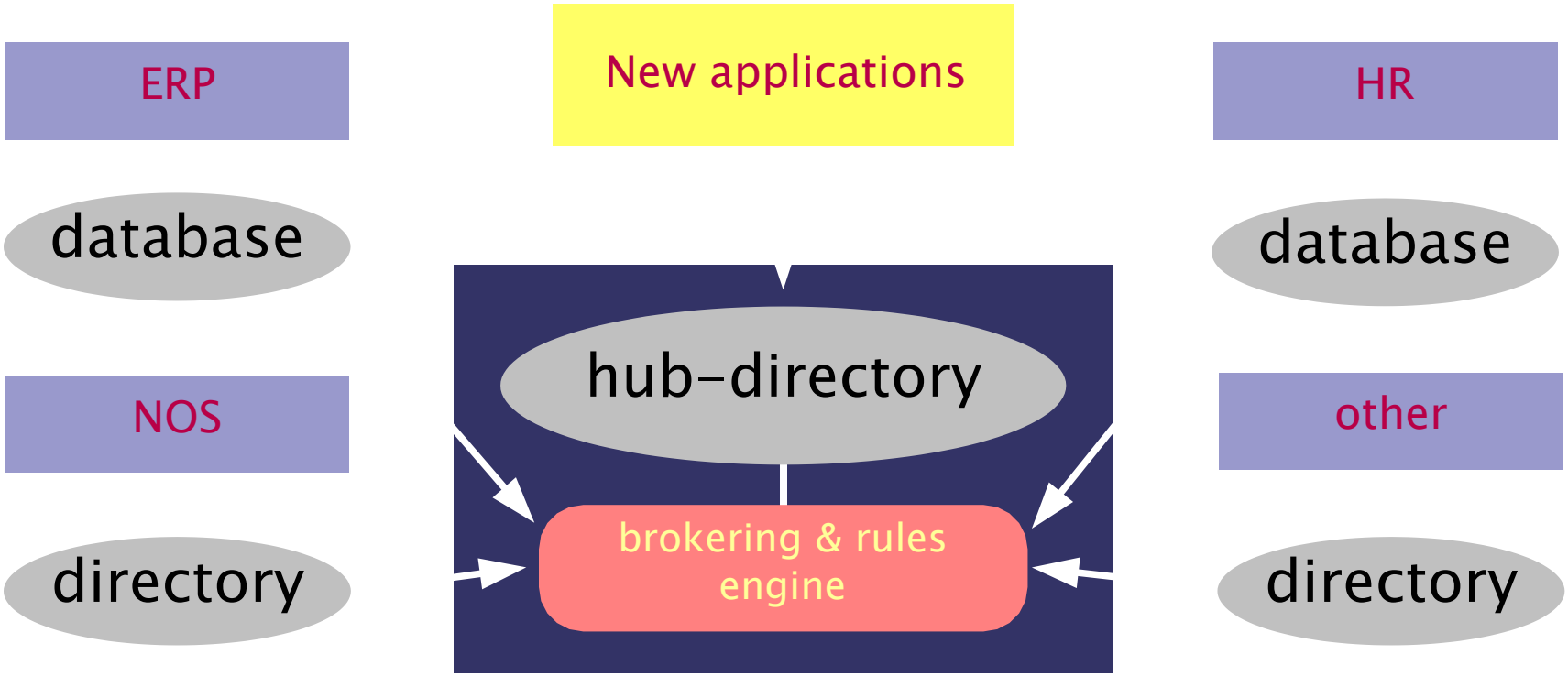
Directory Consolidation

- si modificano tutte le applicazioni in modo che usino lo stesso *directory service*



Hub & Spokes

- le applicazioni si classificano in *legacy* e non-
- ciascuna applicazione accede al rispettivo *repository*, catturando i cambiamenti e propagandoli ai *repository* che devono riflettere il cambiamento



Risultati

Managed identity

Per gli utenti

facile accesso alle applicazioni (*seamless login* o SSO)

stessa *password* per l'accesso ai sistemi

Per la struttura IT

sicurezza applicativa

consistenza e controllo

semplice realizzazione dell'*authorization*

conformità ai requisiti di legge (es. DPS)

Federated Identity Management

Finora abbiamo fatto considerazioni che si applicano solamente al mondo B2E

In scenari B2B o B2C, in cui la gestione delle identità include utenti esterni all'azienda, queste tecniche non sono sufficienti e occorre estenderle

Le Aziende devono gestire le identità non solo dei loro impiegati, ma anche di Clienti, Fornitori e Partner

Le relazioni tra queste identità e il proprio *business* vanno aggiornate di frequente

Federated Identity Management

Di solito non è possibile delegare ad altri questa gestione per vari motivi

- a parte i mercati finanziari *consumer*, non vi sono Identity Providers con tale offerta
- vi sono rischi, di immagine e legali, per il non corretto uso delle informazioni personali associate alle identità esterne

Federated Identity Management

La situazione attuale costringe

- gli utenti a ripetere un'autenticazione ad ogni accesso ad applicazioni esterne all'Azienda
- l'IT a gestire identità, frequentemente variabili nel tempo, che non sono sotto il controllo dell'Azienda

Con la diffusione dei *web services*, la difficoltà diventa sempre maggiore

Federated Identity Management

La tecnica di *federated identity management* consente agli utenti l'accesso trasparente ai siti e alle applicazioni che partecipano alla federazione, senza riautenticarsi

- Esiste un *issuing party* e un *relying party*
- Definisce come si effettua l'*enrollment* e l'*identity linking*
- Definisce come mappare gli attributi delle identità
- Definisce come gestire la *privacy*

Federated Identity Management

issuing party

relying party

0) enrollment

directory

directory

1) authentication

3) request forwarding

2) request to partner portal

company portal

partner portal

5) access to partner portal

4) identity linking

Risultati

managed federated identity

Per gli utenti

Accesso trasparente alle applicazioni fornite dall'esterno

Per la struttura IT

Minori costi di gestione delle identità appartenenti a sistemi esterni

Applicazioni

Come consentire agli utenti l'accesso trasparente a tutte le applicazioni web aziendali (nuove ed esistenti), senza riautenticarsi?

Esistendo l'Identity Management, si può

- definire un Web/Application server principale

autentica l'utente e gli permette il lancio delle applicazioni

- definire tutti i Web/Application server come secondari

essi "ereditano" le credenziali dell'utente già autenticato

- progettare un meccanismo di sessioni federate

che realizzi questo scambio di credenziali in modo trasparente all'utente e consenta ai web/appserver secondari di iniziare la propria sessione web senza chiedere all'utente un'autenticazione

Applicazioni

Come realizzare autorizzazioni *fine-grained* riutilizzando basi dati e logica applicativa esistente?

- Web Services / URLs as auth calls

il protocollo HTTP è disponibile dappertutto e ad esso ci si può appoggiare per effettuare chiamate a logica applicativa remota (anche senza necessariamente realizzare web services…)

- In applicazioni web e client-server

le medesime tecniche risulteranno poi riutilizzabili per applicazioni tradizionali c-s e per applicazioni web

Applicazioni

Fornire agli utenti un unico punto di gestione della password per tutti i sistemi

Loosely integrated directories

ciascuna directory rimane indipendente dalle altre e si sceglie di non realizzare integrazioni forti con l'installazione di un gestore di identità esterno (es. TIM)

- Web/Application server con gestione password

si centralizza l'azione di cambio password nell'ambiente più facilmente accessibile e si diffondono le regole di gestione

- Architettura hub&spokes

l'azione di cambio password può essere automaticamente propagata alle altre directory da automatismi realizzati con chiamate LDAP ad ogni directory esistente