Lightweight Directory Access Protocol (LDAP)

Vivien Boistuaud

UFR Ingénieurs 2000

Filière IR – 3ème année

Plan de l'exposé

1. Les services d'annuaire
2. Les bases de LDAP
3. Mettre en place un annuaire LDAP
4. Exploiter un annuaire LDAP
5. Démonstration

Les services d'annuaire

- Stockage hiérarchique d'informations
- Permet de modéliser des objets:
  - Utilisateurs
  - Matériels Réseaux
  - Groupes
  - Entreprises, Unité Organisationnelle, Pays…
- Et des attributs associés aux objets
  - Texte, données binaires, Digest, Listes…
- Gestion de droits d'accès à l'annuaire (ACL)
- Sécurisé (transport => TLS, authentification => X.509)

Motivations de LDAP

- Besoin d'uniformisation :
  - **Avant**, Un annuaire ⇔ Un protocole d'accès
    - NIS / YellowPages
    - Microsoft SAM (*Security Account Manager*)
    - X.500 DAP jugé trop lourd pour l'implantation
  - **Uniformisation** protocolaire : LDAPv1/v2
    - Au dessus de TCP/IP
    - Communique avec tout service d'annuaire
    - Pas d'impact sur l'implantation de l'annuaire
    - Standardisé en1993 – RFC 1487
      - révisé en 1995 – RFC 1777, puis en 2002 et 2006

LDAP, un standard…

- LDAPv3 en 2002
- LDAPv3 révisé en 2006 (dernière version)
  - RFC 4510 – *Technology Roadmap*
  - RFC 4511 – Le protocole LDAP
  - RFC 4512 – Modèle d'information
  - RFC 4513 – Authentification et Sécurité
  - RFC 4514 – Formatage des *Distinguished Names*
  - RFC 4515 – Formatage des filtres de recherche

…très complet

- Et toujours en 2006
  - RFC 4516 – URLs LDAP
  - RFC 4517 – Syntaxe et Règles de correspondances
  - RFC 4518 – Internationalisation
  - RFC 4519 – Schéma pour les applications utilisateurs

Nombreux aspects

- Pour l'administrateur système
  - Permet de stocker les informations des utilisateurs (login, mot de passe, nom, horaires, home directory, userid…)
  - Authentification uniforme sur le réseau
- Pour le développeur d'applications
  - Authentifier un utilisateur
  - Lire les informations sur une entrée
  - (Eventuellement) associer des informations sur une entrée

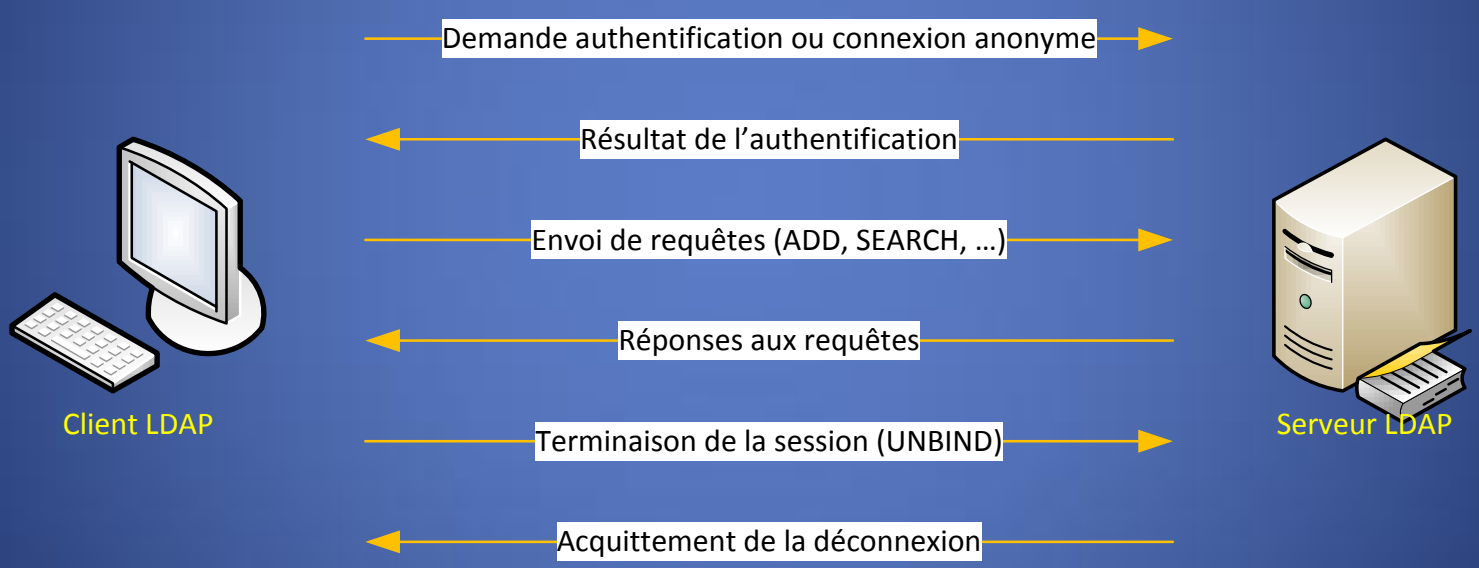
Plan de l'exposé

1. Les services d'annuaire
2. Les bases de LDAP
3. Mettre en place un annuaire LDAP
4. Exploiter un annuaire LDAP
5. Démonstration

Initialement un protocole

- Fonctionne sur TCP/IP
  - Permet de résoudre la problématique du transport et du routage
- Port assigné par l'IANA: 389
  - <1024, réservé au root sous UNIX
- Compatible avec X.500
  - Format d'annuaire normalisé par l'UTI-T
- Format de description uniformisé LDIF
  - *LDAP Data Interchange Format*

Principes de communication



Structuration des informations [1]

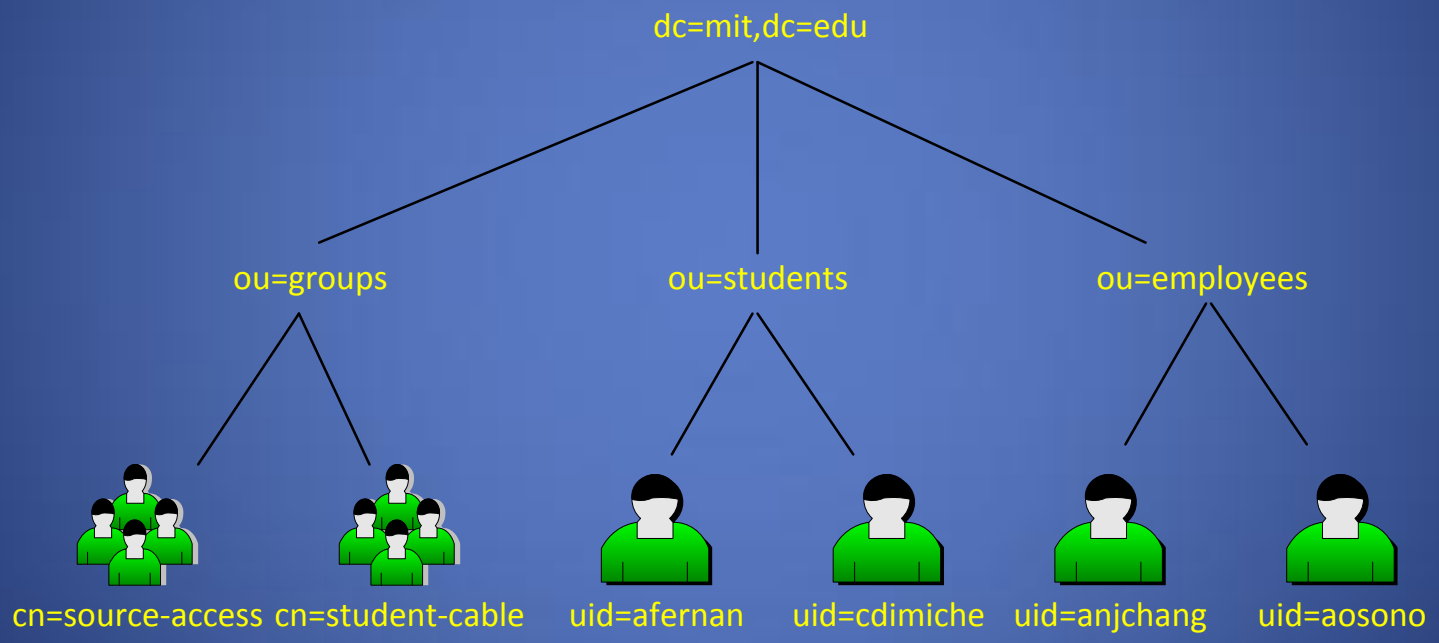
- Définition de schéma
  - Classes d'objets
    - Supporte l'héritage multiple
    - Types de classes : abstract, auxiliary, structural
  - Types d'attributs
    - obligatoires ou optionnels pour une classe
  - Règles de correspondance
    - Moyens de comparaison des données (case sensitive, binaire, hachage…)

Structuration des informations [2]

- Définition de Schéma (suite)
  - Utilisation de règles de correspondances
    - Corrélation règles/utilisation
    - Fonctionnement à la discrétion du fournisseur
  - Syntaxes
    - Contrôle du format des informations
    - Fonctionnement à la discrétion du fournisseur
- Chaque élément de schéma ⇔ OID
- Annuaire = Schéma + Entrées (*Entries*)
  - Comme SGBDR, Schéma ⇔ Entrées
- Schéma ~ f(administrateur, applications)

Stockage des informations [1]

- Entrées hiérarchisées



- Une entrée ⇔ 1..n classes + 1..n attributs
- Ident. Nom distinctif (DN) et format d'URL

Stockage des informations [2]

- Chaque classe peut :
  - Obliger à posséder des attributs (clefs)
  - Permet de gérer des attributs optionnels
    - Si aucune valeur, l'entrée n'est pas stockée
- Chaque attribut
  - Possède un nom (dc, sn, cn, o, ou…)
  - Possède une valeur, avec format définit
  - Peut être de type multiple, read-only, collectif, obsolète (fixé par flags)

Exemple d'entrée (LDIF)

```
dn: cn=Amy V
   Francis+employeeNumber=34f4e35b18979bb6400f3a7b8fa
   88821,ou=employees,dc=mit,dc=edu
objectClass: apple-user
objectClass: eduPerson
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: shadowAccount
objectClass: top
cn: Amy V Francis
gidNumber: 101
homeDirectory: /afs/athena.mit.edu/user/a/m/amyvf
sn: Francis
uid: amyvf
mail: amyvf@mit.edu
o: Massachussetts Institute of Technology
ou: Department of Biological Engineering
title: Financial Assistant II
```

Plan de l'exposé

1. Les services d'annuaire
2. Les bases de LDAP
3. Mettre en place un annuaire LDAP
4. Exploiter un annuaire LDAP
5. Démonstration

Choisir son implantation…

- En fonction de ses besoins !
  - Commercial
    - Microsoft ActiveDirectory (environnement Windows Server / Clients Windows)
    - Oracle Internet Directory
    - Sun Java System Directory Server
    - Novell eDirectory
  - Gratuit et Open-Source
    - OpenLDAP (gratuit & libre)
    - TinyLDAP (gratuit & libre)
    - Apache Directory Server (en Java)

…et les bons outils de travail

- Certains clients gratuits en lecture seule
  - Softerra LDAP Browser
- Souvent des clients spécifiques
  - MMC (Microsoft Management Console)
  - Novell, Sun, Oracle…
- Mais aussi de très bon libres
  - Apache Directory Studio
    - *Basé sur Eclipse Platform*

Ce que l'on souhaite faire

- Stocker des informations
  - Définir les besoins du schéma principal
  - Gérer les droits d'accès (ACL)
- Authentifier les utilisateurs sur le SI
  - Compatible avec le système d'exploitation
  - Compatible avec la norme X.509
- Assurer la disponibilité
  - Load balancing
  - Réplication des informations
- Tout ceci est intégré dans toutes les implantations citées

Définir son schéma

- Souvent, implantation par défaut fournie
  - Respecte les spécifications (uid ⇔ login, cn ⇔ Common Name, dn ⇔ Distinguished Name)
  - Ou pas (SAMaccountName ⇔ login… mais configurable)
- Autoriser les applications à mettre à jour le schéma ?
  - Chez Microsoft, tout le temps (Exchange, SQL Server, …)
  - En général, dépend de la stratégie d'administration

Authentifier les utilisateurs

- Compatibilité avec plusieurs protocoles
  - X.509
  - Kerberos
  - Plaintext, hash (MD5, SHA1…)
- Dépend de l'implantation choisie et du client
- En général: *uid* ⇔ login, *password*
  - Configurable sur certaines implantations

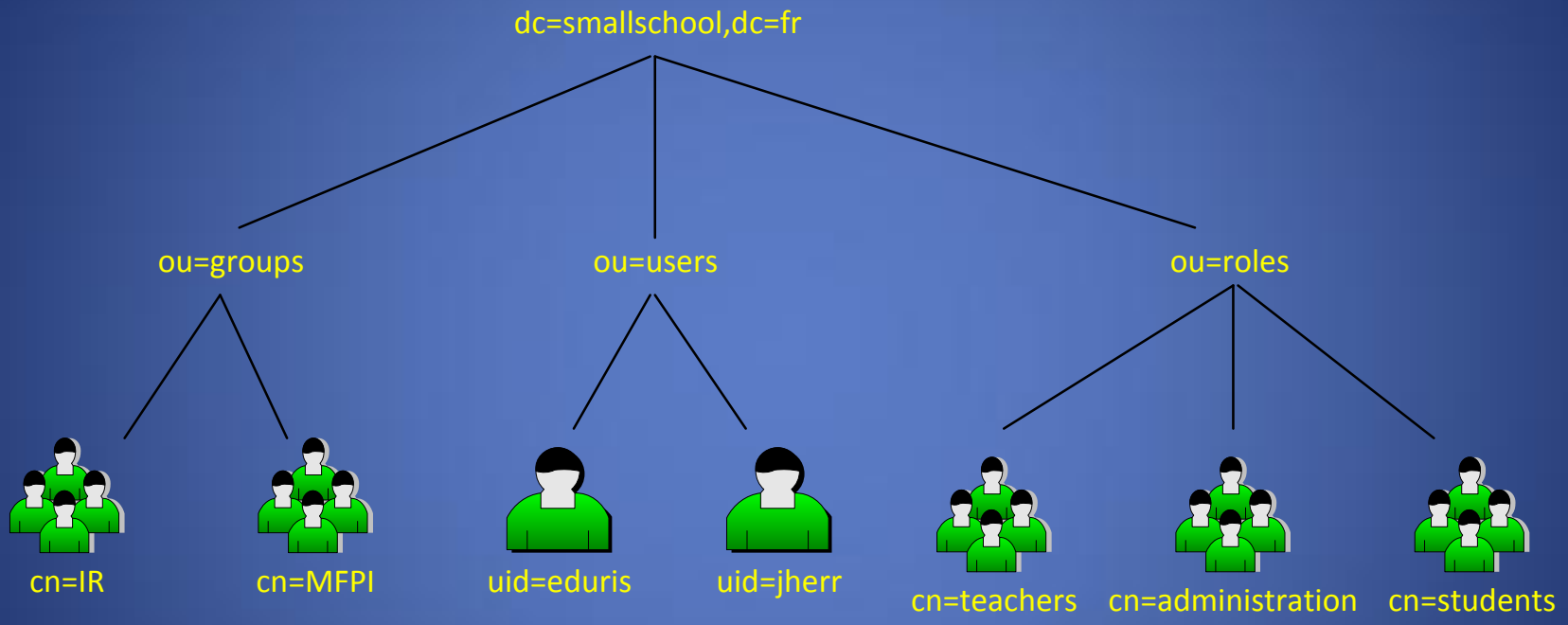
Exemple – SmallSchool [1]

- Choix du serveur d'annuaire
  - Apache Directory Server (portable)
- Choix du client gestion
  - Apache Directory Studio (pratique)
- Environnement
  - 200 étudiants, 50 administratifs, 70 enseignants
  - 2 sites de travail (paris, lyon)
  - Tous sous UNIX/Linux
  - Besoin de compatibilité NIS

Exemple – SmallSchool [2]

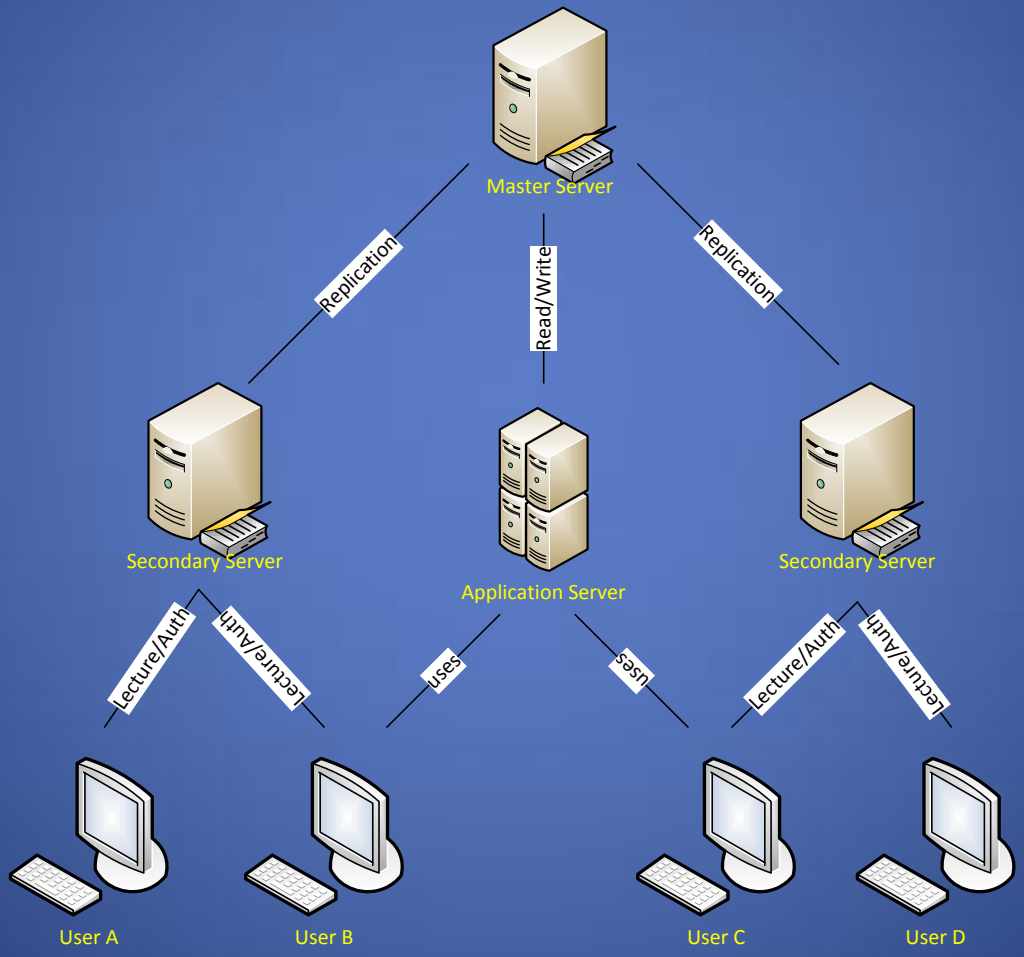
- Création d'un nouvelle instance d'Apache Directory Server
  - Création des fichiers de configuration
  - Création du schéma
    - Importation « *core* », « *system* » et « *nis* »
  - Démarrage de l'instance
    - Sous forme de service windows NT ou
    - Avec la commande suite.exe / suite

Exemple – SmallSchool [3]



- Si nom DNS est ldap.paris.smallschool.fr,
  ldap://ldap.paris.smallschool.fr/cn=IR,ou=groups,dc=smallschool,dc=fr

Exemple - Réplication



Plan de l'exposé

1. Les services d'annuaire
2. Les bases de LDAP
3. Mettre en place un annuaire LDAP
4. Exploiter un annuaire LDAP
5. Démonstration

Pourquoi « parler » LDAP

- Compatibilité avec les services d'annuaires
  - *Tout le monde a un LDAP aujourd'hui*
- Pas besoin d'implanter le protocole
  - Tous les langages parlent LDAP
    - Java (JNDI, jLDAP, SecurityRealms)
    - C, C++
    - C#, VB.NET, J#
    - PHP
    - PERL
    - Ruby, RubyOnRail
    - Python
- Simple d'utilisation (authentification et requêtes)

Le langage de requête LDAP

- (attributeName op. valeurRecherchée)
  - Parenthèses obligatoires
  - Opérations supportées
    - Egalité (=)
    - Supérieur ou égal (>=)
    - Inférieur ou égal (<=)
    - Approximativement (~=)
  - Utilisation possible de *wildcard* (*)

Le langage de requête LDAP

- Possibilité d'utiliser des opérateurs booléens
  - Et (&)
  - Ou (|)
  - Non (!)
  - Exemple :
    - (&(ou=users)(|(cn=Vivien*)(cn=Albert*)))
  - *Par défaut, case insensitive dans beaucoup de cas; dépend de la f() de comparaison du schéma*

Le langage de requête LDAP

- Utilisation des Matching Rules
  - (*attribute*:*matchingRuleName*:=*SeekedValue*)
    - attribute ⇔ nom de l'attribut
    - *matchingRuleName* ⇔ nom règle (ex. *caseExactMatch*)
    - *SeekedValue* ⇔ Valeur recherchée
- Fonctionne aussi par OID
  - Remplacer *:matchingRuleName* par *:dn:OID*
  - Exemple: (sn:dn:2.4.6.8.10:=ExampleSearch)

Plan de l'exposé

1. Les services d'annuaire
2. Les bases de LDAP
3. Mettre en place un annuaire LDAP
4. Exploiter un annuaire LDAP
5. Démonstration

Conclusion

- Ne standardise pas tout
  - Laisse libre de l'implémentation
  - Ne contraint pas les fonctionnements redondés ou la balance de charge
  - Pas nécessairement interopérable entre annuaires
- Unanimement adopté
  - Tous langages, tous fabriquants
- Simple à mettre en œuvre
- Simple d'utilisation

Références

- Technical Roadmap [RFC4510]
- Le protocole LDAP [RFC4511]
- Modèles d'informations [RFC4512]
- Authentification [RFC4513]
- Filtres de recherche [RFC4515]
- URL pour LDAP [RFC4516]
- LDAPBook : http://ldapbook.labs.libre-entreprise.org/book/html/
- LDAP and JNDI together forever : http://www.javaworld.com/javaworld/jw-03-2000/jw-0324-ldap.html