LDAP

**Tài liệu**: [Open Source Guide - LDAP for Rocket Scientists - Contents (zytrax.com)](https://www.zytrax.com/books/ldap/)

Các thuật ngữ quan trọng:

* ou: người thao tác/người sở hữu tài nguyên. Như: ou=people – người sở hữu tài nguyên; ou=groups – nhóm người dùng. Nghĩ nó giống trong linux ý.
* dc=example, dc=com tương ứng với ou=people, dc=example, dc=com.
* sn = last name
* uid = full name

# Sơ lược về LDAP

Trong lịch sử vào những năm cuối 70 – đầu những năm 80, ITU (Liên minh viễn thông quốc tế) đã bắt đầu làm việc trên loạt tiêu chuẩn email X.400. Tiêu chuẩn email này yêu cầu một thư mục tên (và các thông tin khác) có thể được truy xuất trên không gian mạng theo kiểu phân cấp giống với DNS.

Nhu cầu về việc truy cập thư mục trên internet này đã khiến ITU phát triển loạt tiêu chuẩn X.500 (cụ thể là X.519), định nghĩa DAP (Giao thức truy cập thư mục) 🡪 giao thức để truy cập dịch vụ thư mục trên internet

Loạt tiêu chuẩn X.400 và X.500 đi kèm với toàn bộ hệ điều hành OSI, ngốn rất nhiều tài nguyên.

Để rồi đến những năm 90, IETF nhận thấy nhu cầu truy cập vào các thư mục không cần phải chịu tất cả các chi phí tài nguyên lớn khủng khiếp do sử dụng OSI nữa 🡪 bắt đầu sử dụng LDAP. LDAP được thiết kế để cung cấp gần như tất cả các chức năng cơ bản của X.519 nhưng sử dụng giao thức TCP/IP – trong khi vẫn cho phép làm việc với nhau với các thư mục dựa trên X.500.

LDAP khác với DAP ở một số khía cạnh sau:

* Sử dụng TCP/IP, thay vì OSI làm lớp truyền tải mạng.
* Loại bỏ một số chức năng tối nghĩa, trùng lặp.
* Thay thế một số ASN.1 bằng biểu diễn văn bản.

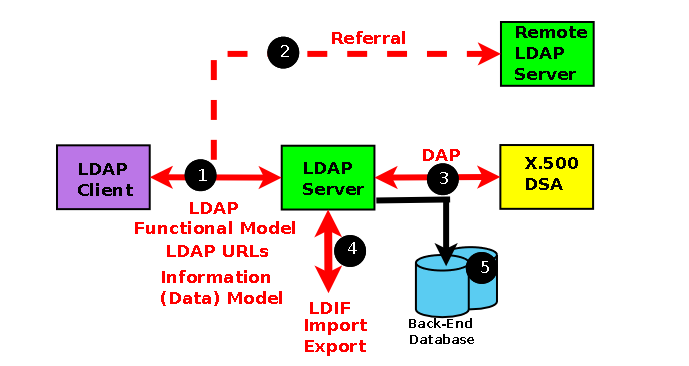
# Tổng quan về LDAP

LDAP là một **giao thức** xác định cách thức mà ‘dữ liệu thư mục’ được truy xuất.

* Nó cũng xác định và mô tả cách dữ liệu được biểu diễn trong dịch vụ thư mục.
* Nó xác định cách dữ liệu được tải và truy xuất từ một dịch vụ thư mục. Chứ không xác định cách dữ liệu được lưu và thao tác.

LDAP xác định bốn mô hình như sau (không liên quan đén hình bên dưới):

1. Mô hình dữ liệu: xác định cách thông tin hoặc dữ liệu được truyền tải trong hệ thống hỗ trợ LDAP.
2. Mô hình đặt tên: Điều này xác định tất cả những thứ `dc=example, dc=com` mà ta bắt gặp trong hệ thống LDAP.
3. Mô hình chức năng: Khi ta đọc, tìm kiếm, viết hoặc sửa đổi LDAP 🡪 đang sử dụng mô hình chức năng.
4. Mô hình bảo mật: Ta có thể kiểm soát một cách rất chi tiết, ai có thể làm gì với dữ liệu nào. Mô hình này cũng bao gồm TLS / SSL.



Giải thích các thành phần trong mô hình trên:

1. LDAP không xác định cách dữ liệu được lưu trữ, chỉ xác định cách nó được truy cập.
2. Khi ta giao tiếp với máy chủ LDAP, ta không cần biết nó đến từ đâu. Về lý thuyết, dữ liệu có thể đến từ 1 hoặc nhiều cơ sở dữ liệu.
3. Ta phải luôn nhớ: quyền truy cập dịch vụ LDAP và hoạt động của dịch vụ LDAP tách biệt. Khi ta thiết kế một hệ thống dựa trên thư mục, hãy xác định nhưng gì ta muốn nó làm (tổ chức dữ liệu) và quên việc triển khai. Sau đó, ở giai đoạn thứ 2, ta mới xác định dữ liệu lấy từ đâu, như thế nào và lưu trữ ở đâu.
4. Một số sản phẩm cơ sở dữ liệu thương mại cung cấp dạng xem LDAP (một trình bao bọc LDAP hoặc một phần trìu tượng LDAP) của các kiểu cơ sơ dữ liệu quan hệ khác. (có nghĩa là có 1 số cơ sở dữ liệu hỗ trợ việc truyền tải dữ liệu bằng LDAP)

# LDAP và Cơ sở dữ liệu

Cơ sở dữ liệu hỗ trợ LDAP được đăc trưng như một dịch vụ ghi một lần, đọc nhiều lần. Điều đó có nghĩa, loại dữ liệu thường được lưu trữ trong dịch vụ LDAP sẽ không thay đổi trong mỗi lần truy cập.

* Ví dụ: LDAP sẽ không phù hợp để duy trì hồ sơ ngân hàng trong mỗi lần truy cập, vì về bản chất chúng thay đổi trong mỗi lần truy cập.
* Tuy nhiên, LDAP rất thích hợp để duy trì thông tin chi tiết về các chi nhánh ngân hàng, giờ mở cửa, nhân viên, …

## Đọc được tối ưu hóa.

Không bao giờ rõ ràng trong cụm từ ghi-một-lần-đọc-nhiều-lần chỉ là bao nhiêu?

Bao giờ ta nên sử dụng cơ sở dữ liệu hỗ trợ LDAP, bao giờ ta nên sử dụng cơ sở dữ liệu quan hệ thông thường như: SQLite, MySql, PostGreSQL. Nếu cứ sau mỗi 2 query, ta lại cập nhật 1 lần thì việc sử dụng LDAP có hợp lý hay không? hay phải là 1000 hoặc 1 triệu lần?

Không có câu trả lời đơn giản cho điều này, nhưng có 1 số lưu ý hữu ích sau:

1. Hiệu suất đọc ghi dữ liệu, phụ thuộc vào việc cập nhật các chỉ mục (index) nào. Càng nhiều chỉ mục (khiến đọc nhanh hơn) nhưng cập nhật càng chậm hơn. Tỷ lệ đọc ghi là 1000:1 và có thể cao hơn đối với các thư mục LDAP được tối ưu hóa để đọc nhiều và 500:1 đối với việc lập chỉ mục để đọc dữ liệu ở mức độ vừa.
2. LDAP Replication tạo ra nhiều ‘giao dich’ cho mỗi bản cập nhật. – bị ảnh hưởng bởi tỷ lệ đọc ghi.
3. Nếu khối lượng dữ liệu lớn (giả sử > 100.000 entry) thì thời gian cập nhật, ngay cả với số lượng chỉ mục nhỏ cũng có thể nghiêm trọng. Vì vậy, ta nên giữ số lượng cập nhật thấp nhất trong thực tế. Tỷ lệ 10.000:1 hoặc cao hơn.
4. Nếu khối lượng dữ liệu tương đối nhỏ (giả sử < 1.000 tạo thành nhiều lần sử dụng LDAP dựa trên bảo mật), các chỉ mục khiêm tốn (tối đa là 2 hoặc 3) và không có bản sao nào đang được sử dụng 🡪 là thích hợp để sử dụng LDAP.

## Khả năng hiển thị của tổ chức dữ liệu

Các phần tử thấp nhất (phần tử giao thức LDAP) được sử dụng để truy cập, các thư mục hỗ trợ LDAP, sử dụng mô hình dữ liệu trừu tượng hóa từ tổ chức vật lý của nó.

* Các phần tử thấp nhất giả định một mô hình dữ liệu đối tượng mà không nhận thức được cấu trúc thực tế của dữ liệu. Việc triển khai máy chủ LDAP cụ thể, trong chức năng back-end của nó, sẽ thực hiện ánh xạ tổ chức dữ liệu vật lý LDAP một cách tự động.
* Điều này hoàn toàn trái ngược với SQL, trong đó các truy vấn SQL được sử dụng để truy vấn dữ liệu, trong đó câu truy vấn có kiến thức đầy đủ và chi tiết về cấu trúc dữ liệu của bảng, CSDL

## Đồng bộ hóa và sao chép dữ liệu

Cơ sở dữ liệu quan hệ có độ dài cực cao để đảm bảo rằng dữ liệu nhất quán trong các chu kỳ ghi/cập nhật bằng các kỹ thuật như: giao dịch (transaction), khóa, rollback và các phương pháp khác. 🡺 Đây là một yêu cầu quan trọng và cần thiết đối với loại CSDL này. Hình thức đồng bộ dữ liệu **cực đoan** này cũng vẫn tồn tại khi dữ liệu được sao chép trên nhiều máy chủ, để đảm bảo dữ liệu luôn nhất quán.

Dữ liệu trong LDAP master và slave sử dụng một quy trình sao chép không đồng bộ đơn giản. Điều này có tác dụng làm cho hệ thống master-slave không đồng bộ hóa dữ liệu trong chu kỳ sao chép. Một truy vấn tới master và slave trong khoảng thời gian này (thường rất ít) có thể mang lại một kết quả không chính xác.

* Điều này có vẻ rất tồi tệ, nhưng ví dụ: 1 nhân viên tên là Bob Smith được hiển thị là trong bộ phận kế toán ở server LDAP và trong bộ phận bán hàng trên máy chủ khác trong vài giây hoặc ít hơn, thì không sao cả.

**Lưu ý:**  Việc triển khai LDAP hiện nay, đặc biệt là những triển khai hỗ trợ cấu hình Đa chủ, ngày càng trở nên phức tạp trong việc sao chép các cập nhật, nhưng tốc độ mạng cao có thể khiến các hoạt động sao chép nhanh hơn đáng kể. 🡺 Tuy nhiên, những vấn đề này chỉ đơn giản là giảm khoản thời gian 2 hệ thống bất kỳ đồng bộ hóa, chứ chúng không loại bỏ hành vi không đồng bộ hóa của LDAP.

## Tóm tắt sử dụng LDAP.

Vậy lợi ích của LDAP là gì?

Trước khi cố gắng trả lời câu hỏi này, ta phải gạt bỏ vấn đề về hiệu suất, bởi hệ thống RDBMS nhanh hơn đáng kể so việc triển khai LDAP. Nhưng khoảng cách về tốc độ này đã đã được giảm bớt trong LDAP thế hệ thứ hai.

Dưới đây là các lý do để ta sử dụng LDAP:

1. LDAP cung cấp phương thức truy cấp dữ liệu cụ bộ và từ xa được **tiêu chuẩn hóa**. Trong khi hầu hết các hệ thống RDBMS cung cấp các phương thức truy cập dữ liệu cục bộ.
2. Vì LDAP sử dụng các phương pháp truy cập dữ liệu được chuẩn hóa, LDAP client và LDAP server có thể phát triển độc lập.
3. LDAP cung cấp một phương pháp, trong đó dữ liệu có thể được di chuyển (ủy quyển) đến nhiều vị trí mà không ảnh hưởng đến bất kỳ quyền truy cập từ bên ngoài nào vào dữ liệu. Bằng cách sử dung các phương pháp giới thiệu, dữ liệu LDAP có thể được chuyển sang các máy chủ LDAP thay thế bằng cách chỉ thay đổi các thông số hoạt động. Do đó, có thể xây dựng các hệ thống phân tán.
4. Hệ thống LDAP có thể được định cấu hình để sao chép dữ liệu tới 1 hoặc nhiều máy chủ hoặc ứng dụng LDAP mà không cần thêm bất kỳ một đoạn mã nào hoặc thay dổi quyền truy cập bên ngoài nào vào dữ liệu đó.

# Mô hình thông tin LDAP (dữ liệu hoặc đối tượng)

Các thư mục được bật LDAP, sử dụng mô hình dữ liệu để đại diện cho dữ liệu dưới dạng phân cấp của các **đối tượng**. 🡺 Điều này không có nghĩa là LDAP là 1 cở sở dữ liệu hướng đối tượng.

* Như đã nói ở trên, bản thân LDAP là một **giao thức** cho phép truy cập vào một dịch vụ hỗ trợ LDAP và không xác định cách dữ liệu được lưu trữ - nhưng các nguyên tắc hoạt động (đọc, xóa, sửa) trên mô hình dữ liệu có đặc điểm giống đối tượng.

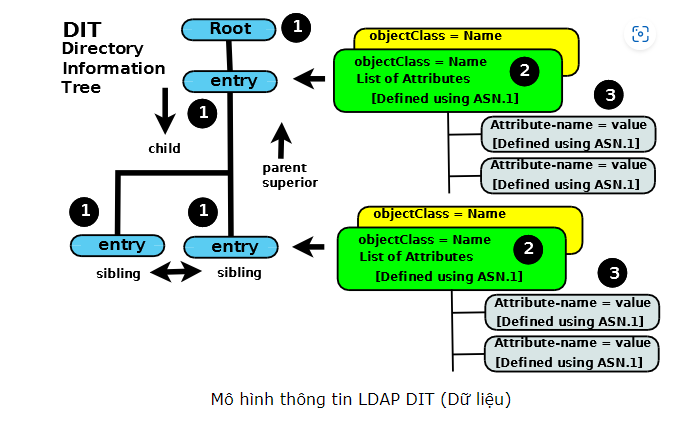
## Cấu trúc cây đối tượng.

Phần này xác định bản chất của LDAP. Là phần quan trọng nhất.

Dữ liệu được biểu diễn trong hệ thống LDAP dưới dạng phân cấp các đối tượng, mỗi đối tượng được gọi là **entry**. Cấu trúc cây được gọi là ‘Cây thông tin thưc mục – Directory Information Tree (DIT)’. Với gốc là **root**.

Mỗi entry bao gồm 1 hoặc nhiều đối tượng. Các đối tượng chứa không hoặc nhiều thuộc tính. Các thuộc tính của chúng được mô tả bằng các định nghĩa ASN.1.

Sơ đồ quan hệ:



## Lớp đối tượng - ObjectClass

Hiện tại hiểu các đối tượng này giống các đối tượng trong java, chỉ khác là được lưu trữ bằng ASN.1

## Các thuộc tính

Các thuộc tính về cơ bản cũng giống Java.

Có các đặc điểm sau:

1. Mỗi thuộc tính xác định kiểu dữ liệu bằng từ khóa SYNTAX.
2. Các thuộc tính cũng có thể kể thừa giống như trong java.
3. Các thuộc tính có thể là tùy chọn (**MAY**) hoặc bắt buộc (**MUST**) giống như trong mô tả ASN.1 của lớp.

* Các thuộc tính có thể là tùy chọn trong đối tượng này nhưng bắt buộc trong đối tượng khác.
* Chính lớp xác định thuộc tính này.

1. Các thuộc tính có thể duy nhất (**SINGLE**) hoặc đa trị (**MULTI-VALUE**) giống như trong mô tả ASN.1 của lớp.

* SINGLE có nghĩa là tất cả các đối tượng có chung 1 giá trị thuộc tính.
* MULTI-VALUE có nghĩa là các đối tượng khác nhau có thể có các giá trị của thuộc tính khác nhau.
* Giá trị mặc định của thuộc tính là MULTI-VALUE.

1. Các thuộc tính có tên và có thể có cả bí danh, tùy theo định nghĩa ASN.1 của nó.

* Ví dụ: thuộc tính **cn** trong **objectClass** là tên người dùng, thuộc tính này có bí danh là **commonName**.

1. Ở mỗi cấp độ phân cấp trong hệ thống cây thông tin, giá trị trong thuộc tính có thể được sử dụng để xác định duy nhất một entry. Hoặc nó có thể đơn giản là thuộc tính như bất kỳ thuộc tính nào khác. Hay giá trị của thuộc tính có thể là sự kết hợp của 2 hay nhiều giá trị thuộc tính khác.

## Mô tả cây bằng cách thêm thông tin vào entry

Việc mô tả cấu trúc cây thông tin và tập hợp dữ liệu ban đầu được thực hiện bằng cách thêm các entry cùng với các đối tượng và thuộc tính của chúng. Điều này bắt đầu từ thư mục gốc 🡪 root luôn được thêm vào đầu tiên trước khi bất kỳ entry nào được thêm vào.

Các entry bao gồm một hoặc nhiều đối tượng, chỉ 1 đối tượng được cấu trúc (STRUCTURAL), không có hoặc có các đối tượng phụ trợ (AUXILIARY). Các thuộc tính chứa dữ liệu không chứa đối tượng.

Trước đây, khi cây thông tin được tạo (DIT), mỗi entry sẽ được nhận dạng duy nhất (so với root) trong hệ thống phân cấp. Để xác định duy nhất một entry, ta cần xác định nội dung dữ liệu của nó. Nội dung dữ liệu được định nghĩa trong thuộc tính (chứa bởi một objectClass) nên ta phải xác định thuộc tính đó là duy nhất (SINGLE).

* Bởi vì thuộc tính có thể có nhiều giá trị, ở mỗi đối tượng 1 thuộc tính có thể có các giá trị khác nhau 🡪 ta cần khai báo rõ ràng các giá trị của thuộc tính. Điều này được thực hiện bằng cách sử dụng **attribute-name**=**value**.
* Ví dụ: thuộc tính duy nhất ở trong entry này là **fred**, là giá trị của thuộc tính **cn** 🡪 ta cần khai báo **cn=fred**.
* Ta cũng có thể viết **cn=fred+drink** để đáp ứng nhu cầu thuộc tính duy nhất trong trường hợp không nghĩ ra giá trị thuộc tính duy nhất.

Việc thêm entry có thể thực hiện theo nhiều cách khác nhau. Một trong số đó là sử dụng Tệp trao đổi dữ liệu LDAP(**LDIF**) được mô tả trong chương sau.

* **LDIF** là các tệp văn bản được sử dụng để mô tả cây thông tin và dữ liệu được thêm vào từng thuộc tính.

Sau đây là ví dụ đơn giản về tệp LDIF thiết lập DN gốc (dc=example, dc=com) và 3 entry con dưới entry people:

version: 1

## version not strictly necessary (and some implementations reject it) but generally good practice

## DEFINE DIT ROOT/BASE/SUFFIX ####

## uses RFC 2377 (domain name) format

## dcObject is an AUXILIARY objectclass and MUST

## have a STRUCTURAL objectclass (organization in this case)

# this is an ENTRY sequence and is preceded by a BLANK line

dn: dc=example,dc=com

dc: example

description: The best company in the whole world

objectClass: dcObject

objectClass: organization

o: Example, Inc.

## FIRST Level hierarchy - people

# this is an ENTRY sequence and is preceded by a BLANK line

dn: ou=people, dc=example,dc=com

ou: people

description: All people in organisation

objectClass: organizationalUnit

## SECOND Level hierarchy - people entries

# this is an ENTRY sequence and is preceded by a BLANK line

dn: cn=Robert Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: Robert Smith

cn: Robert

sn: Smith

uid: rsmith

mail: robert@example.com

mail: r.smith@example.com

ou: sales

## SECOND Level hierarchy - people entries

# this is an ENTRY sequence and is preceded by a BLANK line

dn: cn=Bill Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: Bill Smith

cn: William

sn: Smith

uid: bsmith

mail: bill@example.com

mail: b.smith@example.com

ou: support

## SECOND Level hierarchy - people entries

# this is an ENTRY sequence and is preceded by a BLANK line

dn: cn=John Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: John Smith

sn: smith

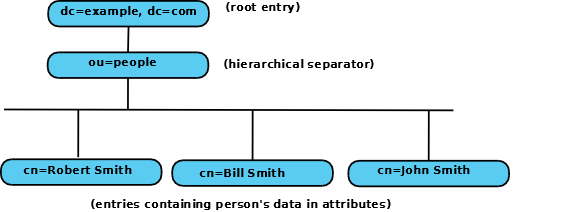
uid: jsmith

mail: jim@example.com

mail: j.smith@example.com

ou: accounting

**Cấu trúc của file LDIF trên**:



**Lưu ý**: Việc thêm các entry có thể được thực hiện bằng ứng dụng LDAP client, chẳng hạn như trình duyệt LDAP hoặc 1 số ứng dụng chuyên dụng.

**Lưu ý quan trọng**: Các dòng trong tệp LDIF ở trên bắt đầu bằng ‘**dn**:’, về cơ bản là cho máy chủ LDAP biết cách cấu trúc hoặc đặt entry trong DIT (sẽ được giải thích thêm ở phần sau).

* Nói chung, giá trị thuộc tính nào được sử dụng cho mục đích này không quan trọng, miễn là ‘dn:’ là duy nhất. 🡪 có nghĩa **dn** dùng để nhận dạng duy nhất một entry.
* Ví dụ: ở file bên trên có “dn: cn=Robert Smith, ou=people, dc=example, dc=com” có thể đổi thành “dn: uid=rsmith, ou=people, dc=example, dc=com”
* dn có thể nhận bất cứ giá trị nào. Những nếu dn được sử dụng cho chức năng đăng nhập thì giá trị của nó là cực kỳ quan trọng và ràng buộc dn trong biệt ngữ thường là **uid**.

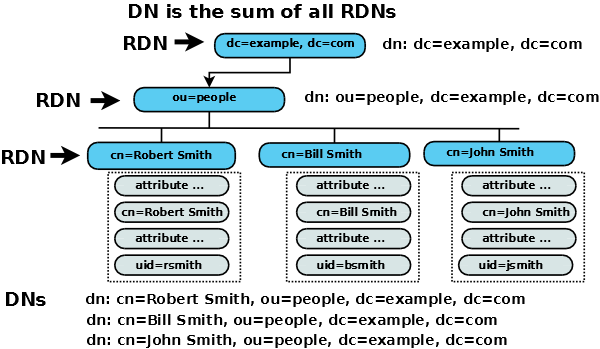
## Thao tác cây

Để có thể thao tác cây thông tin, ta phải gửi các lệnh (đọc, tìm kiếm, sửa đồi, …) đến máy chủ LDAP. Để làm được điều đó ta phải có khả năng cho máy chủ LDAP biết dữ liệu ở đâu (để ghi) hoặc ở vị trí nào (để đọc và tìm kiếm).

Ta có:

* cn=Robert Smith xác định duy nhất một entry là tên phân biệt tương đối RDN.
* còn tên phân biệt DN là tổ hợp của đường dẫn thư mục gốc DIT và tổng tất cả các RDN (được nối với nhau bởi dấu ,).
* Ví dụ: trong ví dụ bên trên tên phân biệt được mô tả bởi AVA cn=Robert Smith đi từ dc=example, dc=com (RDN) đến (RDN) ou=people rồi kết thúc bằng (RDN) cn=Robert Smith.
* Ta có thể sử dụng nhiều AVA để tạo một thuộc tính duy nhất của entry như sau: cn=Robert Smith + uid=rsmith (thường được gọi là RDN đa giá trị). Lúc này DN tương đương của nó sẽ là: cn=Robert Smith + uid=rsmith, ou=people, dc=example, dc=com

Sơ đồ minh họa DN và RDN:



# Ủy quyền và replication LDAP

Một trong những khía cạnh mạnh mẽ hơn của LDAP là thiết kế có khả năng uy thác trách nhiệm bảo trì một phần của thư mục trong khi vẫn xem thư mục là một thể nhất quán.

* Do đó một thư mục của công ty có thể tạo một ủy quyền về trách nhiệm đối với một thư mục của bộ phận cụ thể.
* Về mặt này, LDAP gần như phản ành chính xác khái niệm ủy quyền DNS.

Không giống như DNS, không có tùy chọn tiêu chuẩn nào để yêu cầu máy chủ LDAP tuân theo - ứng dụng LDAP client sẽ trao đổi trực tiếp với máy chủ mới bằng cách sử dụng ủy quyền được trả lại.

* Tương tự, vì tiêu chuẩn không xác định tổ chức dữ liệu LDAP nên nó không trái với tiêu chuẩn đề máy chủ LDAP đi theo liên kết và một số máy chủ LDAP tự động thực hiện chức năng này bằng cách sử dụng quy trình thường được gọi là **chain**.

OpenLDAP xem xét tiêu chuẩn theo nghĩa đen và không xâu chuỗi theo mặc định, nó luôn trả về một lượt ủy quyền. Tuy nhiên, OpenLDAP có thể được cấu hình để cũng cấp **chain** bằng cách sử dụng chỉ thị **chain directive**.

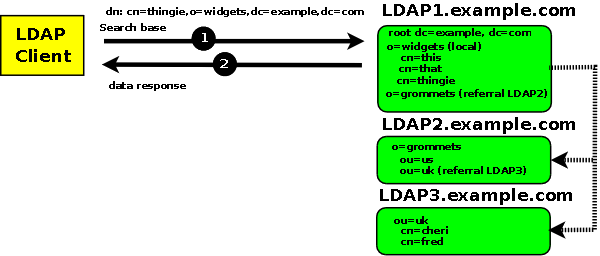
Các tính năng sao chép tích hợp sẵn của LDAP cho phép một hoặc nhiều bản sao của thư mục (DIT) được thay thế từ một bản gốc duy nhất (và thậm chì trong một số triển khai giữa nhiều bản gốc).

Điều quan trọng là phải nhấn mạnh sự khác biệt giữa LDAP và cơ sở dữ liệu. Khi một bản cập nhật được thực hiện trên một thư mục hỗ trợ LDAP master, có thể mất một chút thời gian để cập nhật đến các slave. master-slave (master-master) có thể không đồng bộ trong khoảng thời gian.

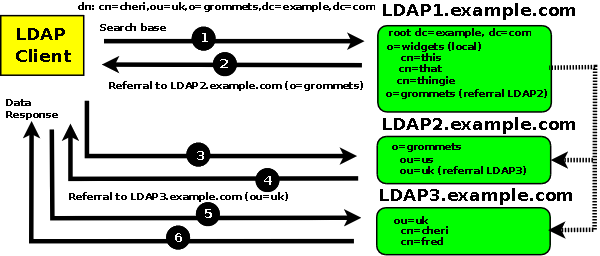
Trong ngữ cảnh LDAP, tạm thời thiếu đồng bộ hóa DIT được coi là không quan trọng. Nhưng trong trường hợp CSDL, ngay cả sự thiếu đồng bộ hóa tạm thời cũng được coi là thảm họa.

## Ủy quyền LDAP

Hình bên dưới hiển thị yêu cầu tìm kiếm với DN cơ bản là dn:cn=thangie,o=widgets,dc=example,dc=com tới hệ thống LDAp dựa trên ủy quyền, yêu cầu này hoàn toàn được đáp ứng từ máy chủ LDAP đầu tiên.



Hình tiếp theo, hiển thị yêu cầu tìm kiếm đến DN cở bản là: dn:cn=cheri,ou=uk,o=gromments,dc=example,dc=com tới hệ thoongsLDAP dựa trên lượt ủy quyền, dẫn đến 1 loạt lượt ủy quyền đến máy chủ LDAP2 và LDAP3, client luôn tuân thao các giới thiệu:



**Ghi chú**:

1. Tất cả các yêu cầu của khách hàng bắt đầu tại thư mục chung LDAP 1.
2. Tại LDAP 1, các yêu cầu đối với bất kỳ dữ liệu nào có ‘widgets’ dưới dạng RDN trong DN đều được đáp ứng ngay lập tức từ LDAP 1, ví dụ: dn:cn=thingie,o=widgets,dc=example,dc=com
3. Tại LDAP, các yêu cầu đối với bất kì dữ liệu nào có ‘grommets’ dưới dạng RDN trong DN đều được chuyển đến LDAP 2, ví dụ: dn:cn=cheri,ou=ok,o=grommets,dc=example,dc=com
4. Tại LDAP 2, các yêu câu đối với bất kỳ dữ liệu nào có ‘uk’ dưới dạng RDN trong DN đều được chuyển đến LDAP 3, ví dụ: dn:cn=cheri,ou=uk,o=grommets,dc=example,dc=com.
5. Nếu máy chủ LDAP được định cấu hình theo **chain** (theo hình 1) thì một phản dữ liệu sẽ được cung cấp cho LDAP client. Chain được kiểm soát bới cấu hình máy chủ LDAP và bởi các giá trị trong yêu cầu tìm kiếm.
6. Các hình minh họa chain rõ ràng bằng cách sử dụng ObjectClass ủy quyền, các máy chủ OpenLDAP có thể được cấu hình để trả về một ủy quyền mặc định nếu không tìm DN mong đợi.

## LDAP Replication

Các tính năng sao chép cho phép sao chép các bản cập nhật LDAP DIT vào một hoặc nhiều hệ thống LDAP vì lý do sao lưu hoặc hiệu suất.

* Nhấm mạnh hoạt động sao chép ở cấp DIT chứ không phải cấp máy chủ LDAP vì có thể có nhiều DIT trong một máy chủ LDAP.
* Sao chép được diễn ra định kỳ ở 1 khoảng thời gian gọi là replication cycle time (về cơ bản là thời gian để gửi dữ liệu đến slave và nhận xác nhận thành công).

🡺Nói chung, có các phương pháp để giảm thời gian của replication cycle time nhưng những phương pháp này thường sẽ có chi phí sử dụng mạng hoặc hiệu năng.

Hai cấu hình sao chép:

1. Master-slave: Trong cấu hình master-slave, một DIT duy nhất được cập nhật và những cập nhật này được sang chép sang một hoặc nhiều slave. DIT trong slave chỉ có thể đọc.

Trong một số điều kiện nhất định, cấu hình slave có thể cung cấp một khả năng cân bằng tải đáng kể. Tuy nhiên cấu hình master-slave có 2 thiếu sót:

* Nếu tất cả người dùng có khả năng đọc/cập nhật DIT thì họ sẽ phải truy cập vào một máy chủ (với DIT slave) để đọc bình thường và một máy chủ khác (với DIT master) để thực hiện cập nhật. Ngoài ra, người dùng có thể trỏ đến master để đọc dữ liệu 🡪 trường hợp này slave hoạt động như 1 bản sao lưu.
* Vì chỉ có 1 master chứa DIT chính nên nó đại diện cho 1 điểm lỗi duy nhất đối với hoạt động ghi (mặc dù DIT slave có thể được định cấu hình để hoạt động như 1 master trong trường hợp xảy ra lỗi lớn).

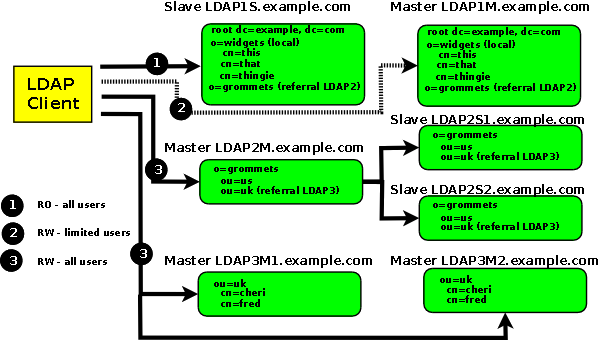
1. Master-master: Trong cấu hình nhiều master, một hoặc nhiều máy chủ chạy DIT chính có thể được cập nhật và kết quả được truyền đến máy chủ ngang hàng.

Trong lịch sử, OpenLDAP không hỗ trợ hoạt động đa chủ trong 1 khoảng thời gian nhưng phiên bản 2.4 đã giới thiệu chức năng đa chủ.

Master-master có thể dẫn đến các tranh chấp cập nhật:

* Tranh chấp giá trị: Nếu 2 cập nhật thuộc tính được thực hiện đồng thời (trong thời gian chu kỳ sao chép) với các giá trị khác nhau, thì tùy thuộc tính là SINGLE hay MULTI-VALUE, mục kết quả có thể ở trạng thái không chính xác hoặc không sử dụng được.
* Tranh chấp xóa: Nếu một người dùng thêm một entry cùng 1 lúc trong chu kỳ sao chép với 1 người khác xóa entry trước đó 🡪 entry đã xóa sẽ xuất hiện lại.

Minh họa:



**Ghi chú**:

1. RO = chỉ đọc, RW: đọc ghi
2. Hệ thống hướng tới máy khách LDAP 1 là 1 slave và ở chế độ chỉ đọc. Người dùng phải đưa ra các sửa đổi/chèn cho master.
3. Hệ thống LDAP 2 đối mặt với máy khách là 1 master và nó được sao chép thành 2 slave.
4. LDAP 3 là 1 master-master và các máy khách có thể đọc/ghi/sửa đổi cho 1 trong 2 master. Mỗi master trong cấu hình này có thể có 1 hoặc nhiều DIT slave.

# Lược đồ LDAP, lớp và thuộc tính.

Khi ta tạo một entry trong DIT, nội dung dữ liệu của nó được chứa trong các thuộc tính, được nhóm thành các lớp (objectclass), được đóng gói thành các lược đồ (schema).

## Tổng quan

Mọi thứ trong LDAP đều có thứ bậc – các lớp và thuộc tính cũng vậy. Các lược đồ cung cấp các đơn vị đóng gói để nhóm các thuộc tính và lớp có liên quan đến nhau.

Các quy tác quan trọng liên quan đến từng vấn đề được xác định bên dưới:

1. Lược đồ: giống package trong java.
2. Lớp - ObjectClass

* Các lớp có thể là STRUCTURAL, trong trường hợp này các lớp này có thể dùng để tạo entry.
* Các lớp có thể là AUXILIARY, các lớp này có thể dùng để thêm vào bất kỳ entry nào.
* Lớp ABSTRACT phổ biến nhất là **top**, tạo thành cấp cao nhất của mọi hệ thống phân cấp lớp và kết thúc bất kỳ hệ thống phân cấp nào.
* Nếu một đối tượng là 1 phần của hệ thống phân cấp (cây kế thừa) thì phải cùng loại (STRUCTURAL, AUXILIARY) với lớp cha. Ngoại lệ đối với quy tắc này là nếu lớp cha là loại ABSTRACT.
* các lớp được định nghĩa bằng ASN.1

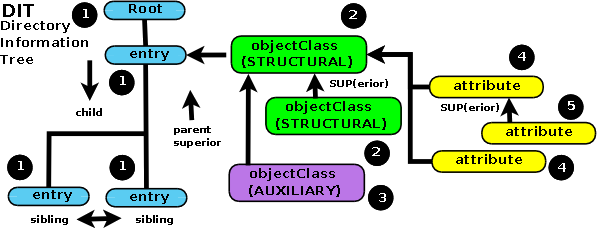
1. Thuộc tính

* Để sử dụng thuộc tính trong một entry, lớp của nó phải được bao gồm trong định nghĩa của entry và lớp của nó phải được bao gồm trong lược đồ.
* Các thuộc tính được định nghĩa bằng ASN.1
* Một thuộc tính bao gồm loại (type) của nó (hoặc SYNTAX), ví dụ: String hoặc number và cách nó hoạt động trong một số điều kiện nhất định, chẳng hạn như liệu các thao tác so sánh có phân biệt chữ hoa chữ thường hay không? phân biệt hoa thường sử dụng quy tắc đối sánh (match rule).

1. Các entry nhóm các lớp trong một DIT:

* các entry phải chứa một và chỉ một lớp STRUCTURAL. Một lớp STRUCTURAL có thể có lớp cha (SUPerior ) cũng là lớp STRUCTURAL 🡪 do đó, cấu trúc phân cấp có thể được xem như 1 lớp STRUCTURAL duy nhất.
* Các entry có thể chứa nhiều lớp AUXILIARY.
* các entry có thể có các entry con.
* các entry có thể có các entry cha.
* các entry có thể có các entry cùng cấp độ với chúng.
* các entry có thể có 3 loại: object entry – chứa dữ liệu của người dùng trong các thuộc tính của lớp, alias entry chứa 1 thuộc tính duy nhất là aliasedObjectName, entry con được sử dụng để lưu trữ dữ liệu quản trị hoặc hoạt động liên quan.

Sơ đồ minh họa:



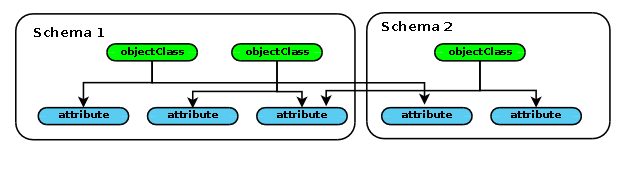
## Lược đồ LDAP

Tao có cảm giá lược đồ LDAP giống lược đồ trong các mô hình dữ liệu quan hệ hơn.

Quy tắc: Mọi thuộc tính hoặc lớp, được sử dụng trong triển khai LDAP phải được xác định trong lược đồ và lược đồ đó phải được khai báo trong cấu hình máy chủ LDAP.

Trong OpenLDAP OLC: **cn**=config, các lược đồ đã cài đặt nằm trong cn=schema, cn=config.

Sơ đồ minh họa:



## Lớp LDAP

Một ObjectClass là một tập hợp các thuộc tính và có các đặc điểm sau:

1. Phải được định nghĩa trong schema
2. Một lớp có thể là một phần của hệ thống phân cấp, trong trường hợp đó nó kế thừa tất cả các thuộc tính của lớp cha.
3. lớp có 1 tên và 1 mã định danh duy nhất.
4. Một lớp, là một bộ chứa thuộc tính, đồng thời cũng là một thuộc tính, có thể xuất hiện trong thao tác tìm kiếm.
5. Một hoặc nhiều lớp phải có trong entry.
6. Mỗi lớp được máy chủ LDAP hỗ trợ tạo thành một phần của bộ sưu tập được gọi là các lớp ‘có thể phát hiện thông qua lược đồ con’.

### Định nghĩa một lớp

Form để tạo một đối tượng:

ObjectClassDescription = "(" whsp

numericoid whsp ; ObjectClass identifier

[ "NAME" qdescrs ]

[ "DESC" qdstring ]

[ "OBSOLETE" whsp ]

[ "SUP" oids ] ; Superior ObjectClasses

[ ( "ABSTRACT" / "STRUCTURAL" / "AUXILIARY" ) whsp ]

; default structural

[ "MUST" oids ] ; AttributeTypes

[ "MAY" oids ] ; AttributeTypes

extensions

whsp ")"

**whsp** ở đây ý là một khoảng trắng (hoặc tab, LF, CR, FF).

Ví dụ:

objectclass ( 2.5.6.2 NAME 'country' DESC 'RFC2256: a country'

SUP top STRUCTURAL

MUST c

MAY ( searchGuide $ description

**Giải thích**:

* 2.5.6.2 NAME ‘country’ là miêu tả về tên toàn cầu của objectClass.
* 2.5.6.2 là OID là mã định danh đối tượng.
* NAME ‘country’ là tên đối tượng.
* SUP ‘top’: ý chỉ đối tượng này có 1 lớp cha là top ~ Object trong java.
* STRUCTURAL chỉ ra rằng lớp này chứa các thuộc tính để tạo thành 1 entry.
* Chỉ có một lớp STRUCTURAL trong một entry nhưng 1 lớp STRUCTURAL có thể xuất hiện trong nhiều entry
* DESC ‘a country’ dùng để mô tả về lớp.
* MUST c: chỉ ra các thuộc tính sau đây là bắt buộc. Trong trường hợp này là c (hoặc countryName) phải có mặt để tạo thành các thể hiện của objectClass. Nếu đây là 1 objectClass STRUCTURAL thì một entry sẽ không được tạo. Nếu chỉ có 1 thuộc tính, ta sẽ viết như ví dụ. Nếu có nhiều thuộc tính, các thuộc tính sẽ được phân tách bằng dấu $. Ví dụ: MUST (a $ b)
* MAY (searchGuide $ description) chỉ ra các thuộc tính sau là tùy chọn, có thể có hoặc không trong việc tạo các thể hiện.

### Một số lớp khác

Đây là cách định nghĩa một lớp ABSTRACT:

objectclass ( 2.5.6.0 NAME 'top' ABSTRACT

MUST objectClass )

Đây là cách định nghĩa lớp AUXILIARY:

objectclass ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'

DESC 'RFC2247: domain component object'

SUP top AUXILIARY MUST dc )

Đoạn sau đây cho thấy định nghĩa DN cơ sở khá điển hình bằng cách sử dụng lớp AUXILIARY trên:

dn: dc=example,dc=com

dc: example.com

objectclass: dcObject

objectclass: organization

o: Example, Inc.

**objectclass: organization** (một lớp STRUCTURAL) tạo entry.

Dưới đây là ví dụ về 1 lớp pilotOrganization kế thừa lớp organization và unitOrganization:

objectClasses: ( 0.9.2342.19200300.100.4.20 NAME 'pilotOrganization'

SUP ( organization $ organizationalUnit ) STRUCTURAL

MAY buildingName )

## Thuộc tính LDAP

Các thuộc tính viết tắt thông dụng:

* commonName: cn
* knownName: gn
* surname: sn

### Xác định 1 thuộc tính

AttributeTypeDescription = "(" whsp

numericoid whsp ; AttributeType identifier

[ "NAME" qdescrs ] ; name used in AttributeType

[ "DESC" qdstring ] ; description

[ "OBSOLETE" whsp ]

[ "SUP" oid ] ; derived from this other

; AttributeType

[ "EQUALITY" woid ; Matching Rule name

[ "ORDERING" woid ; Matching Rule name

[ "SUBSTR" woid ] ; Matching Rule name

[ "SYNTAX" whsp noidlen whsp ] ; Syntax OID

[ "SINGLE-VALUE" whsp ] ; default multi-valued

[ "COLLECTIVE" whsp ] ; default not collective

[ "NO-USER-MODIFICATION" whsp ]; default user modifiable

[ X-ORDERED whsp type ] ; non-standard - default not X-ORDERED

[ "USAGE" whsp AttributeUsage ]; default userApplications

extensions

whsp ")"

Ví dụ:

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

* SUP name: chỉ ra rằng thuộc tính này có thuộc tính cha là **name**.

Ví dụ: định nghĩa thuộc tính name

attributetype ( 2.5.4.41 NAME 'name'

EQUALITY caseIgnoreMatch

SUBSTR caseIgnoreSubstringsMatch

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

* EQUALITY caseIgnoreMatch: cho biết các thuộc tính này sẽ hoạt động trong bộ lọc tìm kiếm. `caseIgnoreMatch` là so sánh không phân biệt hoa thường
* SUBSTR caseIgnoreMatch: cho biết cách bộ lọc xác định chuỗi con. caseIgnoreMatch là so khớp không phân biệt hoa thường.
* SYNTAX 1.3.6.1.4.1.1466.115.121.1.15.121.1.15 {32768} là OID (mã định danh) xác định loại dữ liệu và quy tắc nào được áp dụng cho dữ liệu. Trong trường hợp này OID xác định nó là loại chuỗi có bộ ký tự là ISO 10646. Giá trị {32768} cho biết độ dài tối đa của chuỗi.

Tra cứu: <https://www.zytrax.com/books/ldap/apc/rfc2252.txt> để biết thêm chi tiết.

### Các đặc điểm khác

**SINGLE-VALUE**: mặc định bị tắt 🡪 các thuộc tính có mặc định là MULTI-VALUE.

Để khai báo một thuộc tính là SINGLE-VALUE ta làm như sau:

attributetype ( 0.9.2342.19200300.100.1.25

NAME ( 'dc' 'domainComponent' )

DESC 'RFC1274/2247: domain component'

EQUALITY caseIgnoreIA5Match

SUBSTR caseIgnoreIA5SubstringsMatch

SYNTAX 1.3.6.1.4.1.1466.115.121.1.26

SINGLE-VALUE )

**USAGE** ‘AttributeUsage’ mặc định là **userApplication** (thuộc tính trả về giá trị \* khi tìm kiếm). Giá trị dSAOperation hoặc directoryOperation (trả về giá trị + khi tìm kiếm).

**ORDERING** ‘matchingrule’: quy tắc sắp xếp – thứ tự sắp xếp từ bé đến lớn hoặc ngược lại.

attributetype ( 2.5.4.46 NAME 'dnQualifier'

EQUALITY caseIgnoreMatch

ORDERING caseIgnoreOrderingMatch

SUBSTR caseIgnoreSubstringsMatch

SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )

**COLLECTIVE** – chỉ hợp lệ khi giá trị của USAGE là userApplication 🡪 cho phép các thuộc tính cha của nó chia sẻ cùng một giá trị.

* Quy ước đặt tên: bắt đầu bằng **c-**
* ( 2.5.4.20.1 NAME 'c-TelephoneNumber'
* SUP telephoneNumber COLLECTIVE )

Các thuộc tính COLLECTIVE xuất hiện dưới dạng các entry con của objectClass `collectiveAttributeSubentry`. Bằng cách sửa đổi các thuộc tính COLLECTIVE thông qua các lớp con, tất cả các thuộc tính cha cũng thay đổi.

**NO-USER-MODIFICATTION**: điều này cho biết thuộc tính không được sửa đổi bởi người dùng bất kể mọi quyền được chỉ định. Nó thường được sử dụng khi USAGE là dsaOperation hoặc directoryOperation.

**X-**: bất kỳ đối tường LDAP nào cũng có thể xác định các thuộc tính bổ sung, ngoài các các thuộc tính được khai báo. Tất cả các thuộc tính như vậy phải bắt đầu bằng X- và phải có một tham số được trích dẫn duy nhất. Ví dụ: X-MY-PROPERTY ‘TRUE’

Lệnh X – cái này dành cho những nhà nghiên cứu.

## Quy tắc so sánh (MatchRules)

Quy tắc so sánh xác định các quy tắc so sánh có sẵn trong máy chủ LDAP:

1. Quy tắc so sánh thường được tích hợp sẵn trong máy chủ LDAP và không cần xác định rõ ràng.
2. Quy tắc so sánh dựa vào EQUALITY, SUBSTR, ORDERING của thuộc tính. Nếu như thuộc tính không hỗ trợ việ sử dụng regular expression, thì trong tìm tìm giá trị SUBSTR của thuộc tính sẽ không được xác đinh.

### OpenLDAP tích hợp sẵn MatchRules

Danh sách dưới đây được tìm tháy trong OpenLDAP bằng cách truy vấn lược đồ con như sau:

ldapsearch -H ldap://ldap.example.com -x -s base -b "cn=subschema" "(objectclass=\*)" matchingrules

# matchingrules may be changed to

# attributetypes objectclasses etc., etc.

Lệnh ‘ldapsearch … matchingrules’ phải viết trên cùng một dòng.

Thay thế ldap.example.com thành tên máy chủ LDAP của ta.

Nếu máy chủ LDAP triển khai trên local thì không cần khai báo máy chủ và đối số -H.

-x cho biết ta không sử dụng bảo mật SASAl và được yêu cầu từu OpenLDAP 2.x

Lệnh trên sẽ trả về một thông tin giống như sau:

# Subschema

dn: cn=Subschema

matchingRules: ( 2.5.13.0 NAME 'objectIdentifierMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

matchingRules: ( 2.5.13.1 NAME 'distinguishedNameMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

matchingRules: ( 2.5.13.2 NAME 'caseIgnoreMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

matchingRules: ( 2.5.13.3 NAME 'caseIgnoreOrderingMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

matchingRules: ( 2.5.13.4 NAME 'caseIgnoreSubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )

matchingRules: ( 2.5.13.5 NAME 'caseExactMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

matchingRules: ( 2.5.13.6 NAME 'caseExactOrderingMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

matchingRules: ( 2.5.13.7 NAME 'caseExactSubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )

matchingRules: ( 2.5.13.8 NAME 'numericStringMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.36 )

matchingRules: ( 2.5.13.10 NAME 'numericStringSubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )

matchingRules: ( 2.5.13.13 NAME 'booleanMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )

matchingRules: ( 2.5.13.14 NAME 'integerMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

matchingRules: ( 2.5.13.15 NAME 'integerOrderingMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

matchingRules: ( 2.5.13.16 NAME 'bitStringMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 )

matchingRules: ( 2.5.13.17 NAME 'octetStringMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )

matchingRules: ( 2.5.13.18 NAME 'octetStringOrderingMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )

matchingRules: ( 2.5.13.20 NAME 'telephoneNumberMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )

matchingRules: ( 2.5.13.21 NAME 'telephoneNumberSubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )

matchingRules: ( 2.5.13.23 NAME 'uniqueMemberMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.34 )

matchingRules: ( 2.5.13.27 NAME 'generalizedTimeMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )

matchingRules: ( 2.5.13.28 NAME 'generalizedTimeOrderingMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )

matchingRules: ( 2.5.13.29 NAME 'integerFirstComponentMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

matchingRules: ( 2.5.13.30 NAME 'objectIdentifierFirstComponentMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

matchingRules: ( 2.5.13.34 NAME 'certificateExactMatch'

SYNTAX 1.2.826.0.1.3344810.7.1 )

matchingRules: ( 1.3.6.1.4.1.1466.109.114.1 NAME 'caseExactIA5Match'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

matchingRules: ( 1.3.6.1.4.1.1466.109.114.2 NAME 'caseIgnoreIA5Match'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

matchingRules: ( 1.3.6.1.4.1.1466.109.114.3 NAME 'caseIgnoreIA5SubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

matchingRules: ( 1.3.6.1.4.1.4203.1.2.1 NAME 'caseExactIA5SubstringsMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

matchingRules: ( 1.2.840.113556.1.4.803 NAME 'integerBitAndMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

matchingRules: ( 1.2.840.113556.1.4.804 NAME 'integerBitOrMatch'

SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

Để biết thông tin của từng `matchingRules` ta tra cứu: [OID assignments from the top node (alvestrand.no)](https://www.alvestrand.no/objectid/top.html)

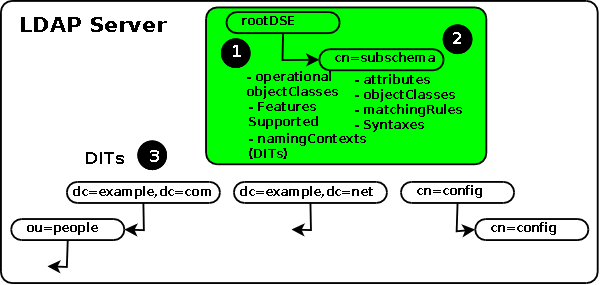
## LDAP Opeational attibutes and object

Có một loại các thuộc tính và lớp được tích hợp trong máy chủ LDAP chi phối cách thức hoạt động của nó.

* Các thuộc tính và lớp này thường được gọi **opeational**.

Tất cả những thứ này đều nằm trong rootDSE và không xuất hiện trong các hoạt động bình thường.

Mối quan hệ giữa các DIT và các entry của nó với RootDSE được hiển thị bên dưới:



RootDSR có thể được kiểm tra bằng lệnh sau:

ldapsearch -H ldap://ldap.example.com -x -s base -b "" +

# note the + returns operational attributes

Ta có thể sử dụng trang web: [OID assignments from the top node (alvestrand.no)](https://www.alvestrand.no/objectid/top.html) để tra cứu các thông tin tiện ích.

### SubSchema subentry objects

Subschema subentry có thể được tra cứu bằng lệnh sau:

ldapsearch -H ldap://ldap.mydomain.com -x -s base -b "cn=subschema" objectclasses

# the list of attributes that may be listed are

# matchingruleuse ldapsyntaxes matchingrules attributetypes

# the above entries are collections

# createtimestamp modifytimestamp

# if you use + alone you will get a huge list of

# everything the LDAP server knows about.

Lệnh trên liệt kê tất cả các lớp được khai báo trong máy chủ LDAP.

# Cài đặt LDAP

Có 2 cách để cài đặt LDAP:

1. Sử dụng ApacheDS
2. Sử dụng OpenLDA

## Hướng dẫn cài đặt OpenLDAP

### Ldap Server

OpenLDAP chỉ hỗ trợ hệ điều hành linux, không hỗ trợ window.

Tham khảo cài đặt: [Install and Configure OpenLDAP Server on Ubuntu 22.04|20.04|18.04 | ComputingForGeeks](https://computingforgeeks.com/install-and-configure-openldap-server-ubuntu/)

Và [LDAP Server - LDAP Account Manager - Ubuntu Server 18.04 - YouTube](https://www.youtube.com/watch?v=7BoD1pC-_LM&t=475s)

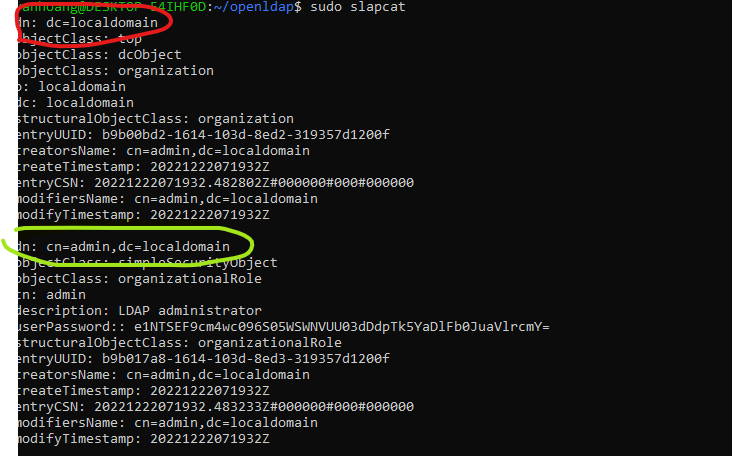
Lệnh cài đặt: sudo apt -y install slapd ldap-utils

Thư mục cài đặt: /etc/ldap

Khai báo LDAP server trong /etc/hosts. Ví dụ: 192.168.8.58 ldap.server.com (Bước này chỉ cần thiết khi muốn thay đổi domain của ldap server )

Để xác nhận đã cài đặt thành công hay chưa: sudo slapcat – để hiển thị DIT.

Thông tin hiển thị ra như sau:



* Bôi màu đỏ, chính là root, cũng là tên của phân vùng dữ liệu. Ta không thể sử dụng entry này để thêm các entry vào DIT.
* Bôi màu xanh, là tài khoản admin, có vài trò thêm, xóa, sửa các entry khác.

**Ví dụ**:

Tạo 2 entry cơ bản, đại diện cho ‘chủ sở hữu’ và ‘nhóm người dùng’

nano basedn.ldif

dn: ou=people,dc=localdomain

objectClass: organizationalUnit

ou: people

dn: ou=groups, dc=localdomain

objectClass: organizationalUnit

ou: groups

Chạy lệnh sau để thêm entry:

ldapadd –x –D cn=admin,dc=localdomain –W –f basedn.ldif

**Ví dụ**:

Thêm 1 entry đại diên cho người dùng:

dn: uid=vanhoang,ou=people,dc=localdomain

objectClass: inetOrgPerson

objectClass: posixAccount

objectClass: shadowAccount

cn: vanhoang

sn: hoang

userPassword: {SSHA}Zn4/E5f+Ork7WZF/alrpMuHHGufC3x0k

loginShell: /bin/bash

uidNumber: 2000

gidNumber: 2000

homeDirectory: /home/vanhoang

Giá trị của thuộc tính `userPassword` có thể nhận được bằng lệnh:

sudo slappasswd

**Ví dụ**:

Thêm 1 entry trong ‘nhóm người dùng’:

$ vim ldapgroups.ldif

dn: cn=hoangnv,ou=groups,dc=localdomain

objectClass: posixGroup

cn: hoangnv

gidNumber: 2000

memberUid: vanhoang

### LDAP Client

Tham khảo: [Configure LDAP Client on Ubuntu 22.04|20.04|18.04|16.04 | ComputingForGeeks](https://computingforgeeks.com/how-to-configure-ubuntu-as-ldap-client/)

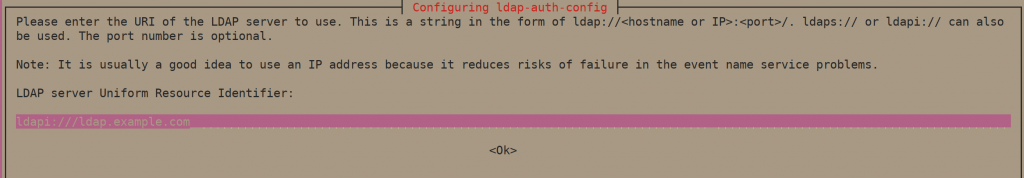
**Chú ý**: Ldap client là ám chỉ các ứng dụng kết nối đến ldap server để kiểm tra thông tin. Vì vậy ứng dụng Spring Security cũng được coi là ldap client.

Lệnh cài đặt LDAP Client: sudo apt –y install libnss-ldap libpam-ldap ldap-utils

File cấu hình: /etc/ldap.conf

**Chú ý**:

1. Tại màn cấu hình khi cài đặt:



Thông tin điền vào là địa chỉ LDAP server khai báo trong /etc/hosts. Ví dụ:

ldap://127.0.0.1:389 – server mặc đinh trong local

1. `searchbase` nhập root trong slapcat. Ví dụ: dc=localdomin
2. version = 3
3. maker local root database admin: yes
4. does the ldap database required login: no
5. ldap account for root: Nhập admin trong slapcat. Ví dụ: cn=admin,dc=localdomain
6. ldap root account password: Nhập password của ldap server.

Các bước sau, tùy ý, không cần thực hiện cũng được.

**Chú ý**: Tài khoản người dùng ở đây là tài khoản người dùng của linux. Sau khi cài đặt xác thực thành công, kiểm tra bằng cách gọi lệnh: **id**

Sau khi cài đặt thành công, ta thêm xác thực ldap bằng cách sửa file: /etc/nsswitch.conf như sau:



Sau đó, sửa file /etc/pam.d/common-password: - xóa `use\_authtok` của dòng trông như sau:

password [success=1 user\_unknown=ignore default=die] pam\_ldap.so use\_authtok try\_first\_pass

Cài đặt để cho phép tạo thư mục chính trong lần đăng nhập đâu tiên bằng cách thêm dòng sau vào cuối file /etc/pam.d/common-session

session optional pam\_mkhomedir.so skel=/etc/skel umask=077

### LDAP Account Manager

**Các bước**:

1. sudo apt -y install apache2 php php-cgi libapache2-mod-php php-mbstring php-common php-pear
2. sudo a2enconf php\*-cgi
3. sudo service apache2 reload
4. sudo apt -y install ldap-account-manager
5. sudo nano /usr/share/ldap-account-manager/config/lam.conf

Nội dung file như sau:

# LDAP Account Manager configuration

#

# Please do not modify this file manually. The configuration can be done completely by the LAM GUI.

#

###################################################################################################

# server address (e.g. ldap://localhost:389 or ldaps://localhost:636)

ServerURL: ldap://localhost:389

# list of users who are allowed to use LDAP Account Manager

# names have to be seperated by semicolons

# e.g. admins: cn=admin,dc=yourdomain,dc=org;cn=root,dc=yourdomain,dc=org

thay bằng admin trong slapcat

Admins: cn=admin,dc=localdomain

# password to change these preferences via webfrontend (default: lam)

mật khẩu tùy chỉnh

Passwd: yenbinh

# suffix of tree view

# e.g. dc=yourdomain,dc=org

thay bằng root trong slapcat

treesuffix: dc=localdomain

# default language (a line from config/language)

defaultLanguage: en\_GB.utf8

# Path to external Script

scriptPath:

# Server of external Script

scriptServer:

# Access rights for home directories

scriptRights: 750

# Number of minutes LAM caches LDAP searches.

cachetimeout: 5

# LDAP search limit.

searchLimit: 0

# Module settings

modules: posixAccount\_user\_minUID: 10000

modules: posixAccount\_user\_maxUID: 30000

modules: posixAccount\_host\_minMachine: 50000

modules: posixAccount\_host\_maxMachine: 60000

modules: posixGroup\_group\_minGID: 10000

modules: posixGroup\_group\_maxGID: 20000

modules: posixGroup\_pwdHash: SSHA

modules: posixAccount\_pwdHash: SSHA

# List of active account types.

activeTypes: user,group

thay bằng entry trong slapcat có objectclass: inetOrgPerson. Nếu chưa có thì tạo mới

types: suffix\_user: ou=people,dc=localdomain

types: attr\_user: #uid;#givenName;#sn;#uidNumber;#gidNumber

types: modules\_user: inetOrgPerson,posixAccount,shadowAccount

thay bằng entry trong slapcat có objectclass: posixGroup

types: suffix\_group: ou=groups,dc=localdomain

Cuối cùng đăng nhập vào ldap-account-manager bằng cách truy cập vào: 127.0.0.1/lam. Nếu không được thì kiểm tra: slapd và apache2 đã bật chưa?

## Hướng dẫn cài đặt ApacheDS

Tải apacheds server và apache directory studio tại:[Welcome to ApacheDS — Apache Directory](https://directory.apache.org/apacheds/)

Ví dụ: ApacheDS 2.0.0.AM26 và Apache Directory Studio 2.0.0-M17

**Ghi chú**:

Với Apache server ta có:

* Nơi cài đặt hệ thống là: C:\Program Files (x86)\ApacheDS
* Vị trí file log: C:\Program Files (x86)\ApacheDS\instances\default\log
* Nếu hệ thống không khởi động được do lỗi: java.nio.file.NoSuchFileException: C:\Windows\system32\config\systemprofile\AppData\Local\Temp\tempks15822298910020502630ks

🡪 vào thư mục C:\Windows\system32\config\systemprofile\AppData\Local tạo thư mục Temp

Với Apache Directory studio ta có:

* Nơi cài đặt hệ thống là: C:\Program Files\Apache Directory Studio
* Nếu xảy ra lỗi: version 1.8.0\_202 of the JVM is not suitable for this product. Version: 11 or greater is required 🡪 thì vào file: C:\Program Files\Apache Directory Studio\ ApacheDirectoryStudio.ini thêm 3 dòng sau:

-vm

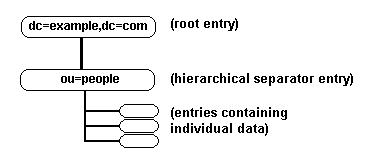
C:\Program Files\Java\jdk-11.0.16.1\bin\javaw.exe

-vmargs

# Ví dụ sử dụng OpenLDAP

## Các ví dụ đơn giản

### Thiết kế DIT



### Chọn STRUCTURAL objectClass

### slapd.conf File

Dưới đây là một file slapd.conf mẫu mô tả cách gán Database cho LDAP, ở đây là Oracle Berkeley Database (BDB) (cơ sở dữ liệu được đề xuất trước đây của OpenLDAP – hiện tại là mdb)

#

###### SAMPLE 1 - SIMPLE DIRECTORY ############

#

# NOTES: inetorgperson picks up attributes and objectclasses

# from all three schemas

#

# NB: RH Linux schemas in /etc/openldap

#

include /usr/local/etc/openldap/schema/core.schema

include /usr/local/etc/openldap/schema/cosine.schema

include /usr/local/etc/openldap/schema/inetorgperson.schema

# NO SECURITY - no access clause

# defaults to anonymous access for read

# only rootdn can write

# NO REFERRALS

# DON'T bother with ARGS file unless you feel strongly

# slapd scripts stop scripts need this to work

pidfile /var/run/slapd.pid

# enable a lot of logging - we might need it

# but generates huge logs

loglevel -1

# MODULELOAD definitions

# not required (comment out) before version 2.3

moduleload back\_bdb.la

# NO TLS-enabled connections

# backend definition not required

#######################################################################

# bdb database definitions

#

# replace example and com below with a suitable domain

#

# If you don't have a domain you can leave it since example.com

# is reserved for experimentation or change them to my and inc

#

#######################################################################

database bdb

suffix "dc=example, dc=com"

# root or superuser

rootdn "cn=jimbob, dc=example, dc=com"

rootpw dirtysecret

# The database directory MUST exist prior to running slapd AND

# change path as necessary

directory /var/db/openldap/example-com

# Indices to maintain for this directory

# unique id so equality match only

index uid eq

# allows general searching on commonname, givenname and email

index cn,gn,mail eq,sub

# allows multiple variants on surname searching

index sn eq,sub

# sub above includes subintial,subany,subfinal

# optimise department searches

index ou eq

# if searches will include objectClass uncomment following

# index objectClass eq

# shows use of default index parameter

index default eq,sub

# indices missing - uses default eq,sub

index telephonenumber

# other database parameters

# read more in slapd.conf reference section

cachesize 10000

checkpoint 128 15

**Ghi chú**:

1. Ta có thể tùy chỉnh các tham số của cơ sở dữ liệu BDB. Giá trị `cachesize` thể hiện kích thước bộ nhớ đệm, `checkpoint` cung cấp điểm phục hồi dữ liệu nếu như CSDL bị lỗi.
2. Các `index` đã tối ưu hóa một số hình thức truy cập nhất định. Ta vẫn có thể tìm kiếm các thuộc tính không được lập chỉ mục, nhưng sẽ mất nhiều thời gian hơn.
3. Bảo mật được xác định bằng cách sử dụng ‘chỉ thị truy cập’ (access directive ). Sự vắng mặt của chúng trong trường hợp này cho phép đọc ẩn danh (không cần xác thực).
4. Không có ‘chỉ thị phụ trợ’ (back-end directive), chỉ thị này không cần thiết.

### Tệp LDIF

LDIF dưới đây sẽ tạo cấu trúc DIT và thêm một entry people.

LDIF sẽ được thêm vào bằng cách sử dụng tiện ích ldapadd của OpenLDAP (slapd) đang chạy.

Duyệt hệ thống phân cấp lớp `inetOrgPerson` sẽ tìm thấy các MUST entry. Sau khi duyệt ta thấy có 2 thuộc tính bắt buộc là **cn** và **sn**.

## DEFINE DIT ROOT/BASE/SUFFIX ####

## uses RFC 2377 format

## replace example and com as necessary below

## or for experimentation leave as is

## dcObject is an AUXILLIARY objectclass and MUST

## have a STRUCTURAL objectclass (organization in this case)

# this is an ENTRY sequence and is preceded by a BLANK line

dn: dc=example,dc=com

dc: example

description: My wonderful company as much text as you want to place

in this line up to 32K continuation data for the line above must

have <CR> or <CR><LF> i.e. ENTER works

on both Windows and \*nix system - new line MUST begin with ONE SPACE

objectClass: dcObject

objectClass: organization

o: Example, Inc.

## FIRST Level hierarchy - people

## uses mixed upper and lower case for objectclass

# this is an ENTRY sequence and is preceded by a BLANK line

dn: ou=people, dc=example,dc=com

ou: people

description: All people in organisation

objectclass: organizationalunit

## SECOND Level hierarchy

## ADD a single entry under FIRST (people) level

# this is an ENTRY sequence and is preceded by a BLANK line

# the ou: Human Resources is the department name

dn: cn=Robert Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: Robert Smith

cn: Robert J Smith

cn: bob smith

sn: smith

uid: rjsmith

userpassword: rJsmitH

carlicense: HISCAR 123

homephone: 555-111-2222

mail: r.smith@example.com

mail: rsmith@example.com

mail: bob.smith@example.com

description: swell guy

ou: Human Resources

**Ghi chú**:

1. Phải có ít nhất một dòng trống giữa các entry (và bắt đầu bằng dn:)
2. Trong nhiều tài liệu, ta sẽ thấy một entry xác định cho `rootdn` (siêu người dùng) trong tệp LDIF (trong ví dụ slapd.conf ta có rootdn “cn=jimbob, dc=example, dc=com”). Chừng nào `rootpw` còn được cung cấp trong tệp slapd.conf thì điều này không cần thiết và có khả năng gây nguy hiểm vị nó để lộ entry này cho bên ngoài truy cập 🡪 điều này là không nên.
3. Ta đã sử dụng nhiều giá trị **cn** để tối đa hóa khá năng tìm thấy người đó theo tên. Ta đã sử dụng index eq,sub (trong tệp slapd.conf) trên thuộc tính cn.

### Tải file LDIF

Để tải tệp LDIF, ta cần một máy chủ OpenLDAP đang chạy, vì vậy để khởi động máy chủ ta làm như sau:

[redhat] /etc/rc.d/init.d/ldap start

[bsd] /usr/local/etc/openldap/slapd.sh start

# confirm slapd is running

ps ax | grep slapd

# (you should see the slapd process entry if it has been started successfully)

Giả sử ta lưu LDIF trên dưới định dạng createdit.ldif trong thư mục /tmp của mình, ta tải tệp LDIF bằng cách sử dụng lệnh `ladpadd` như sau:

ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"

-f /tmp/createdit.ldif -w dirtysecret

-x cho biết ta không sử dụng bảo mật SASAl và được yêu cầu từu OpenLDAP 2.x

-D là ràng buộc và được yêu cầu vì ta đã xác định `rootdn` và không có câu lệnh truy cập nào khác (siêu người dùng có thể làm mọi thứ).

-w là mật khẩu

### Thêm entry bằng LDIF

Ví dụ:

version: 1

## ADD a single entry to people level

dn: cn=John Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: John Smith

cn: John J Smith

sn: Smith

uid: jsmith

userpassword: jSmitH

carlicense: HISCAR 124

homephone: 555-111-2223

mail: j.smith@example.com

mail: jsmith@example.com

mail: john.smith@example.com

ou: Sales

## ADD another single entry to people level

dn: cn=Sheri Smith,ou=people,dc=example,dc=com

objectclass: inetOrgPerson

cn: Sheri Smith

sn: smith

uid: ssmith

userpassword: sSmitH

carlicense: HERCAR 125

homephone: 555-111-2225

mail: s.smith@example.com

mail: ssmith@example.com

mail: sheri.smith@example.com

ou: IT

Giả sử, ta lưu LDIF trên tại file addentry.ldif, trong thư mục /tmp, ta tải tệp LDIF bằng cách sử dụng `ldapadd` như sau:

ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"

-f /tmp/addentry.ldif -w dirtysecret

### Sửa đổi entry

version: 1

## MODIFY the Robert Smith entry

dn: cn=Robert Smith,ou=people,dc=example,dc=com

changetype: modify

add: telephonenumber

telephonenumber: 555-555-1212

telephonenumber: 212

-

replace: uid

uid: rjosmith

-

replace: mail

mail: robert.smith@example.com

mail: bob.smith@example.com

-

# adds using URL format

add: jpegphoto

jpegphoto: < file://path/to/jpeg/file.jpg

-

delete: description

Thực thi tệp kia bằng cách:

ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"

-f /tmp/modentry.ldif -w dirtysecret

## Ví dụ bảo mật

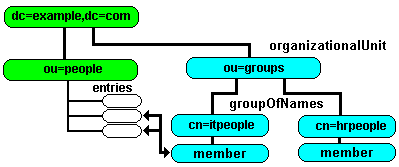
### Chính sách bảo mật

Bây giờ ta sẽ thêm một số bảo mật đơn giản bằng cách chỉnh sửa tệp slapd.conf.

LDAP xây dựng chính sách kiểm soát truy cập (ACP hay còn gọi là ACL) như sau:

1. Chủ sở hữu entry có thể xem và cập nhật tất cả các thuộc entry bao gồm cả mật khẩu.
2. Nhóm người dùng có thể cập nhất bất kỳ entry nào nhưng không được đọc hoặc ghi mật khẩu người dùng. (các entry ou=people là member của entry ou=group).
3. Không ai có thể đọc các entry ‘carlicence’, ‘homepostaddress’, ‘homephone’ ngoại trừ chủ sở hữu và nhóm người dùng.
4. Tất cả người dùng phải xác thực (không cho phép truy cập ẩn danh).
5. Admin phải có khả năng cập nhật hoặc thay đổi mật khẩu trên tất cả entry.

Sơ đồ mình họa:



### Thêm nhóm

Ví dụ sau cho thấy cách thêm nhóm và các entry của nó bằng LDIF:

version: 1

# create FIRST Level groups branch

dn: ou=groups,dc=example,dc=com

objectclass:organizationalunit

ou: groups

description: generic groups branch

# create the itpeople entry under groups

dn: cn=itpeople,ou=groups,dc=example,dc=com

objectclass: groupofnames

cn: itpeople

description: IT security group

member: cn=William Smith,ou=people,dc=example,dc=com

# create the hrpeople entry under groups

dn: cn=hrpeople,ou=groups,dc=example,dc=com

objectclass: groupofnames

cn: hrpeople

description: Human Resources group

member: cn=Robert Smith,ou=people,dc=example,dc=com

**Ghi chú**:

* member: cn=William Smith,ou=people,dc=example,dc=com hiện tại không tồn tại trong DIT. Điều này hoàn toàn chấp nhận được bởi không có kiểm tra nào được thực hiện khi thêm `member`. Hạn chế duy nhất là hiện tại không có thành viên nào trong nhóm **itpeople**.

Ta có thể thêm file LDIF bằng lệnh sau:

ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"

-f /tmp/addgroups.ldif -w dirtysecret

### ACL – truy cập slapd.conf

File mẫu slapd.conf ở bên trên đã được sửa đổi để thêm chình sách kiểm soát truy cập ACL như sau:

#

###### SAMPLE 2 - DIRECTORY with ACL ############

#

# NOTES: inetorgperson picks up attributes and objectclasses

# from all three schemas

#

# NB: RH Linux schemas in /etc/openldap

#

include /usr/local/etc/openldap/schema/core.schema

include /usr/local/etc/openldap/schema/cosine.schema

include /usr/local/etc/openldap/schema/inetorgperson.schema

# NO REFERRALS

# DON'T bother with ARGS file

# pidfle allows scripts for stopping slapd to work

pidfile /var/run/slapd.pid

# enable a lot of logging - we might need it

loglevel -1

# NO dynamic backend modules

# NO TLS-enabled connections

#######################################################################

# bdb database definitions

#

# replace example and com below with a suitable domain

#

# If you don't have a domain you can leave it since example.com

# is reserved for experimentation or change them to My and inc

#######################################################################

database bdb

suffix "dc=example, dc=com"

# ACL1

access to attrs=userpassword

by self write

by anonymous auth

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com"

write

by \* none

# ACL2

access to attrs=carlicense,homepostaladdress,homephone

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"

write

by \* none

# ACL3

access to \*

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"

write

by users read

by \* none

# root or superuser

rootdn "cn=jimbob, dc=example, dc=com"

rootpw dirtysecret

# The database directory MUST exist prior to running slapd AND

# change path as ncessary

directory /var/db/openldap/example-com

# Indices to maintain for this directory

# required if searches will use

# unique id so equality match only

index uid eq

# allows general searching on commonname, givenname and email

index cn,gn,mail eq,sub

# allows multiple variants on surname searching

index sn eq,sub

# sub above includes subintial,subany,subfinal

# optimise department searches

index ou eq

# if searches will include objectClass uncomment following

# index objectClass eq

# shows use of default index parameter

index default eq,sub

# indices missing - uses default eq,sub

index telephonenumber

# other database parameters

# read more in slapd.conf reference section

cachesize 10000

checkpoint 128 15

**Lưu ý**: Các thuộc tính `carlicense`, `hometelephone` và `homePostalAddress` không cần thiết phải xuất hiện trong bất kỳ entry nào. Điều này cho ta thấy rằng:

1. Chính sách bảo mật ACL và nội dung của DIT là độc lập với nhau.
2. Vì 3 thuộc tính trên là một phần của hệ thống phân cấp **inetOrgPerson** objectClass (🡪 organizationalPerson 🡪 Person), ta có thể thêm bất kỳ entry nào trong tương lại mà không cần phải xác định chính sách bảo mật ACL ngay từ đầu.

Bây giờ ta cần dừng OpenLDAP và khởi động lại slapd.conf này:

# stop and start OpenLDAP (slapd)

# on Linux/Redhat

/etc/rc.d/init.d/ldap restart

# on BSD

[bsd] /usr/local/etc/rc.d/slapd.sh stop

# then

[bsd] /usr/local/etc/rc.d/slapd.sh start

# confirm slapd is running

ps ax | grep slapd

## Ví dụ hệ thống phân cấp mở rộng

### Yêu cầu mở rộng

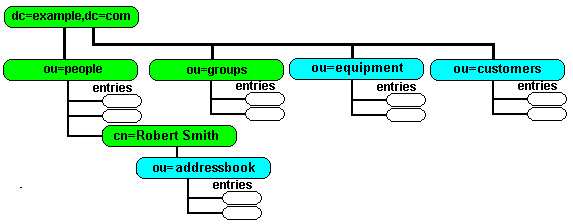
Việc mở rộng như sau:

1. ou=customers – có thể được đọc bởi bất kỳ người dùng (ou=people) xác thực nào nhưng chỉ có thể được thêm vào bởi objectclass có cn=salepeople.
2. equipment (PC, máy in, v.v) sẽ được lưu trữ trong entry và chỉ có thể được cập nhật bởi ou=itpeople.
3. addressbook sẽ được thêm vào danh bạ. Các entry có thể được đọc, tạo và viết bởi cá nhân. Nhưng người dùng sẽ không thể xóa entry addressbook. Chỉ có ou=itpeople mới có thể xóa
4. ou=hrpeople sẽ chịu trách nhiệm thêm và xóa entry của tất cả các nhóm.

Phân quyền như trên bằng cách sử dung ACL trong file slapd.conf.

Còn file LDIF chỉ đóng vai trò khởi tạo dữ liệu.

Sơ đồ:



### Các bước triển khai

Bao gồm các bước sau:

1. Thêm 1 nhóm customers vào DIT. Nhóm sẽ sử dụng **inetorpersion** objectClass.
2. Thêm 1 ou=equipment vào DIT. Nhóm sẽ sử dụng **device** objectClass
3. Thêm 1 nhóm salepeople vào ou=groups.
4. Thêm 1 nhóm addressbook vào ou=people. Nhóm sẽ sử dụng inetorperson objectClass.

### File .ldif

version: 1

# create FIRST level customers branch

dn: ou=customers,dc=example,dc=com

objectclass: organizationalunit

ou: customers

description: customer address book branch

# create FIRST level equipment branch

dn: ou=equipment,dc=example,dc=com

objectclass: organizationalunit

ou: equipment

description: IT assets branch

# create an entry under equipment entry under groups

dn: cn=LP1,ou=equipment,dc=example,dc=com

objectclass: device

cn: LP1

description: Some brand of printer

serialnumber: 1-77-23-15

l: Room 17

owner: cn=John Smith,ou=people,dc=example,dc=com

ou: printers

# create the salespeople entry under groups

dn: cn=salespeople,ou=groups,dc=example,dc=com

objectclass: groupofnames

cn: salespeople

description: Sales group

member: cn=John Smith,ou=people,dc=example,dc=com

# create the addressbook entry for each person

dn: ou=addressbook,cn=John Smith,ou=people,dc=example,dc=com

objectclass: organizationalunit

ou: addressbook

description: Personal Address Book

dn: ou=addressbook,cn=Sheri Smith,ou=people,dc=example,dc=com

objectclass: organizationalunit

ou: addressbook

description: Personal Address Book

dn: ou=addressbook,cn=Robert Smith,ou=people,dc=example,dc=com

objectclass: organizationalunit

ou: addressbook

description: Personal Address Book

### Phân quyền

#

###### SAMPLE 3 - DIRECTORY with enhanced ACL ############

#

# NOTES: inetorgperson picks up attributes and objectclasses

# from all three schemas

# devices objectclass is in core.schema

#

# NB: RH Linux schemas in /etc/openldap

#

include /usr/local/etc/openldap/schema/core.schema

include /usr/local/etc/openldap/schema/cosine.schema

include /usr/local/etc/openldap/schema/inetorgperson.schema

# NO REFERRALS

# DON'T bother with ARGS file

# pidfle allows scripts for stopping slapd to work

pidfile /var/run/slapd.pid

# enable a lot of logging - we might need it

loglevel -1

# NO dynamic backend modules

# NO TLS-enabled connections

#######################################################################

# bdb database definitions

#

# replace example and com below with a suitable domain

#

# If you don't have a domain you can leave it since example.com

# is reserved for experimentation or change them to My and inc

#######################################################################

database bdb

suffix "dc=example, dc=com"

# ACL Notes

# The following additional notes apply for 2.4:

# 1. attrs is now used instead of attr (to reduce warning messages)

# 2. Removed the ,expand modifier with all regex expressions since

# 2.4 rejected some (but not all) ACL's which contained this modifier

# 3. 2.4 checking is much more rigorous and rejected ACL 8 since it contained

# attributes to be added later

# 4. If exact or base contains a regular expression substitution then

# the expand keyword must be used

# ACL1

access to attrs=userpassword

by self write

by anonymous auth

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by \* none

# ACL2

# allow read of addressbook by owner and itpeople; no-one else see it

access to dn.regex="^ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=entry

by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" read

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users none

# ACL3

# allows itgroup to create addressbook but not see entries

access to dn.regex="cn=[^,]+,ou=people,dc=example,dc=com$"

attrs=children

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users none break

# ACL4

# allows creation of entries in own addressbook; no-one else can

# access it, needs write access to the ENTRY attribute (ACL5 or ACL6A)

# and the entries CHILDREN (ACL4)

access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=children

by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write

by users none

# ACL5 - only required prior to 2.2

# allow creation of entries in own addressbook; no-one else can

# access it, needs write access to the ENTRY attribute (ACL5 or ACL6A)

# and the entries CHILDREN (ACL4)

#access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

# attrs=entry

# by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write

# by users none

# ACL6 - only required prior to 2.2

# allow creation of entries in own addressbook; no-one else can

# access it

#access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

# filter=(objectclass=inetorgperson)

# by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write

# by users none

# ACL6A - 2.2+ replace both ACL5 and ACL6 with this ACL

access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=entry,@inetorgperson

by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write

by users none

# ACL7

# allows sales to create entries in customers

# authenticated user can only read

access to dn.one="ou=customers,dc=example,dc=com"

attrs=children

by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write

by users read

# ACL8

access to attrs=carlicense,homepostaladdress,homephone

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write

by \* none

# ACL8A - control access to equipment

access to dn.one="ou=equipment,dc=example,dc=com"

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users read

by \* none

# ACL9

access to \*

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write

by users read

by \* none

# root or superuser

rootdn "cn=jimbob, dc=example, dc=com"

rootpw dirtysecret

# The database directory MUST exist prior to running slapd AND

# change path as ncessary

directory /var/db/openldap/example-com

# Indices to maintain for this directory

# unique id so equality match only

index uid eq

# allows general searching on commonname, givenname and email

index cn,gn,mail eq,sub

# allows multiple variants on surname searching

index sn eq,sub

# sub above includes subintial,subany,subfinal

# optimise department searches

index ou eq

# if searches will include objectClass uncomment following

# index objectClass eq

# shows use of default index parameter

index default eq,sub

# indices missing - uses default eq,sub

index telephonenumber

# other database parameters

# read more in slapd.conf reference section

cachesize 10000

checkpoint 128 15

## Tạo đối tượng

Trong phần này, ta sẽ tạo 1 số thuộc tính, một đối tượng và một lược đồ để đóng gói.

### Yêu cầu

Bổ sung thêm những thứ sau vào các ví dụ trên:

1. Thêm thuộc tính `dohicky`, thuộc kiểu boolean. Và chỉ có thể được đọc và sửa đổi bởi người sở hữu và ‘hrpeople’.
2. Thêm thuộc tính `ageAtBirth`, thuộc kiểu number. Chỉ có thể đọc và sửa đổi bởi người sở hữu và ‘hrpeople’.
3. Thêm thuộc tính `gobbledegook`, thuộc kiểu string và có thể được đọc bởi bất kỳ người dùng nào đã xác thực. Chỉ có thể bị sửa đổi bởi người sở hữu và ‘hrpeople’. Và cho phép lọc bằng <= và >=.
4. Là một phần trong quá trình chuyển sang đăng nhập một lần (SSO) (Xem chương tiếp theo), ta sẽ thêm lớp posixAccount cho mỗi người dùng. Bản ghi sẽ hiển thị với ‘itpeople’.

### Định nghĩa thuộc tính

**dohicky –** boolean

attributetype ( 1.3.6.1.4.1.6863.2.3.107 NAME 'dohicky'

EQUALITY booleanMatch

SYNTAX [1.3.6.1.4.1.1466.115.121.1.7](https://www.zytrax.com/books/ldap/apa/types.html#boolean) )

**ageAtBirth** – number

attributetype ( 1.3.6.1.4.1.6863.2.3.108 NAME 'ageAtBirth'

EQUALITY integerMatch

ORDERING integerOrderingMatch

SYNTAX [1.3.6.1.4.1.1466.115.121.1.27](https://www.zytrax.com/books/ldap/apa/types.html#numbers) )

**gobbledegook –** String

attributetype ( 1.3.6.1.4.1.6863.2.3.109 NAME 'gobbledegook'

EQUALITY caseIgnoreMatch

SUBSTR caseIgnoreSubstringsMatch

ORDERING caseIgnoreOrderingMatch

SYNTAX [1.3.6.1.4.1.1466.115.121.1.44{200}](https://www.zytrax.com/books/ldap/apa/types.html#strings) )

### Định nghĩa ObjectClass và Schema

objectclass ( 1.3.6.1.4.1.6863.2.4.57 NAME 'ourObject'

DESC 'A very useful object'

SUP top AUXILIARY

MUST ( dohicky $ gobbledegook )

MAY ageAtBirth

### Thay đổi file slapd.conf

File slapd.conf sửa đổi như sau:

#

###### SAMPLE 4 - DIRECTORY with home-grown objectclass & attributes ############

#

# NOTES: inetorgperson picks up attributes and objectclasses

# from all three schemas

# devices objectclass is in core.schema

# posixAccount is in nis.schema

# add ourco.schema

#

# NB: RH Linux schemas in /etc/openldap

#

include /usr/local/etc/openldap/schema/core.schema

include /usr/local/etc/openldap/schema/cosine.schema

include /usr/local/etc/openldap/schema/inetorgperson.schema

include /usr/local/etc/openldap/schema/nis.schema

include /usr/local/etc/openldap/schema/ourco.schema

# NO REFERRALS

# DON'T bother with ARGS file

# pidfle allows scripts for stopping slapd to work

pidfile /var/run/slapd.pid

# enable a lot of logging - we might need it

loglevel -1

# NO dynamic backend modules

# NO TLS-enabled connections

#######################################################################

# bdb database definitions

#

# replace example and com below with a suitable domain

#

# If you don't have a domain you can leave it since example.com

# is reserved for experimentation or change them to My and inc

#######################################################################

database bdb

suffix "dc=example, dc=com"

# ACL1

# only owner and itgroup can see or update

access to attrs=userpassword

by self write

by anonymous auth

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by \* none

# ACL1A

# only itgroup can see and update

access to attrs=homedirectory,uidnumber,gidnumber,loginshell,gecos

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by \* none

# ACL2

# allow read of addressbook by owner and itpeople; no-one else see it

access to dn.regex="^ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=entry

by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" read

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users none

# ACL3

# allows itgroup to create addressbook but not see entries

access to dn.regex="cn=[^,]+,ou=people,dc=example,dc=com$"

attrs=children

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users none break

# ACL4

# allow user to create entries in own addressbook; no-one else can access it

# needs write access to the entries ENTRY attribute (ACL5)

# and the entries CHILDREN (ACL4)

access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=children

by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write

by users none

# ACL5

# allow one to create entries in its own addressbook; no-one else can access it

# needs write access to the entries ENTRY attribute (ACL5)

# and the entries CHILDREN (ACL4)

access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

attrs=entry

by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write

by users none

# ACL6

# allow access to all entries in own addressbook; no-one else can access it

access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

filter=(objectclass=inetorgperson)

by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write

by users none

# LDAP 2.2+ replace ACL5 and ACL6 with

#access to dn.regex="ou=addressbook,cn=([^,]+),ou=people,dc=example,dc=com$"

# attrs=entry,@inetorgperson

# by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write

# by users none

# ACL7

# allows sales to create entries in customers

# authenticated user can only read

access to dn.one="ou=customers,dc=zytrax,dc=com"

attr=children

by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write

by users read

# ACL8

access to attr=carlicense,homepostaladdress,homephone,dohicky,ageatbirth

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write

by \* none

# ACL8A - control access to equipment

access to dn.one="ou=equipment,dc=example,dc=com"

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write

by users read

by \* none

# ACL9

access to \*

by self write

by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write

by users read

by \* none

# root or superuser

rootdn "cn=jimbob, dc=example, dc=com"

rootpw dirtysecret

# The database directory MUST exist prior to running slapd AND

# change path as ncessary

directory /var/db/openldap/example-com

# Indices to maintain for this directory

# unique id so equality match only

index uid eq

# allows general searching on commonname, givenname and email

index cn,gn,mail eq,sub

# allows multiple variants on surname searching

index sn eq,sub

# sub above includes subintial,subany,subfinal

# optimise department searches

index ou eq

# if searches will include objectClass uncomment following

# index objectClass eq

# shows use of default index parameter

index default eq,sub

# indices missing - uses default eq,sub

index telephonenumber

# other database parameters

# read more in slapd.conf reference section

cachesize 10000

checkpoint 128 15

**Ghi chú**:

1. Tại đây ta đã thêm các lược đồ: nis.schema (cho posixAccount) và ourco.schema (cho ourObject và các thuộc tính khác).
2. Ở đây ta sử dụng 1 access directive (ACL1A) để giới hạn truy cập vào nhóm ‘itgroup’ của các thuộc tính ‘homedirectory’, ‘uidnumber’, ‘gidnumber’.
3. Ta đã thêm thuộc tính `dohickey` và `ageAtBirth` vào access directive (ACL8) để giới hạn quyền truy cập trong ‘hrpeople’ và chủ sở hữu.

### LDIF

Các entry cũ cập nhật posixAccount objectclass và các thuộc tính MUST trong ourObject objectclass

version: 1

dn: cn=john smith,ou=people,dc=example,dc=com

changetype: modify

objectclass: inetorgperson

objectclass: posixaccount

uidnumber: 200

gidnumber: 203

homedirectory: /var/mail/example.com/jsmith

objectclass: ourObject

dohicky: false

gobbledegook: john

ageatbirth: 0

dn: cn=sheri smith,ou=people,dc=example,dc=com

changetype: modify

objectclass: inetorgperson

objectclass: posixaccount

uidnumber: 201

gidnumber: 203

homedirectory: /var/mail/example.com/ssmith

objectclass: ourObject

dohicky: true

gobbledegook: sheri

dn: cn=robert smith,ou=people,dc=example,dc=com

changetype: modify

objectclass: inetorgperson

objectclass: posixaccount

uidnumber: 202

gidnumber: 203

homedirectory: /var/mail/example.com/rsmith

objectclass: ourObject

dohicky: false

gobbledegook: robert

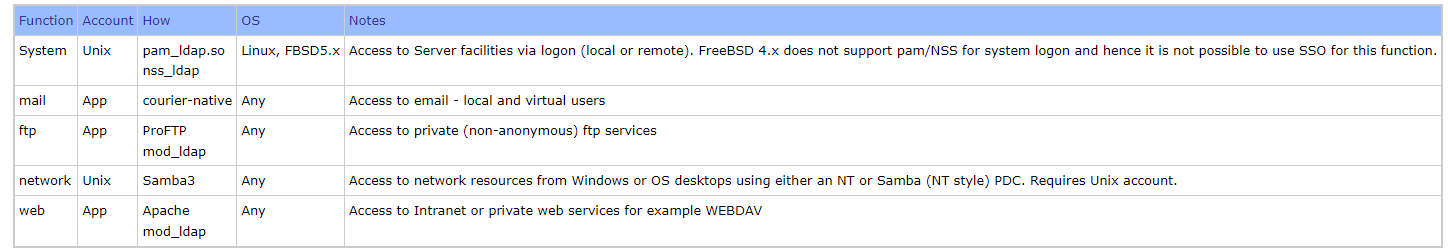
ageatbirth: 17

**Lưu ý**: Trong tất cả thử nghiệm ở trên, ta sẽ thấy trong LDAP gui hiển thị các thông tin của các ou=customer, ou=equiment, ou=groups và những entry của chúng.

* Nếu như không thể nhìn thấy các nhánh (branch) trên, ta phải **dn** của người dùng hoặc **root dn** từ ou=people, dc=example, dc=com thành dc=example, dc=com.

### Đăng nhập một lần (SSO)

Bảng sau đây cho các ví dụ về việc Đăng nhập một lần (SSO) được ứng dụng như thế nào trong các hệ điều hành và không gian mạng:



**Các chức năng của hệ thống đăng nhập một lần**:

1. Một mật khẩu người dùng duy nhất để kiểm soát quyền truy cập vào tất cả các tính năng cho phép người dùng hoặc nhóm người dùng. Điều này không nhất thiết rằng:” một mật khẩu người dùng sẽ cho phép truy cập tất cả các dịch vụ nhưng cho phép một mật khẩu có thể lấy được tất cả các mật khẩu được lưu trữ bởi LDAP”. Người dùng thì chỉ nhìn thấy đăng nhập một lần, trong khi nhiều mật khẩu được cung cấp cho các hệ thống khác nhau.
2. Mật khẩu được lưu ở một vị trí duy nhất để quản trị viên kiểm soát/xác định quyền truy cập vào các tính năng được cho phép cho từng người dùng hoặc nhóm người dùng.
3. Cho phép khả năng tạo, sửa đổi hoặc xóa người dùng và quyền truy cập của người dùng.

# Cấu hình ApacheDS

**Lưu ý**:

* Trong các ví dụ bên dưới: ‘trước khi khởi động máy chủ’ 🡪 là trước khi khởi động máy chủ lần đầu tiên
* ‘máy chủ đang chạy’ 🡪 các lần khởi động máy chủ sau lần đẩu tiên.

## Cấu hình cơ bản

### Thay đổi cổng cho máy chủ LDAP

Theo mặc định, máy chủ LDAP lắng nghe trên công 10389 (không được mã hóa hoặc starttls) và 10636 (SSL)

* việc chạy LDAP trên cổng 389 khá phổ biến, đây là cổng thông dụng nhất của giao thức này, nhưng điều đó yêu cầu máy chủ phải được khởi động với người dùng root.

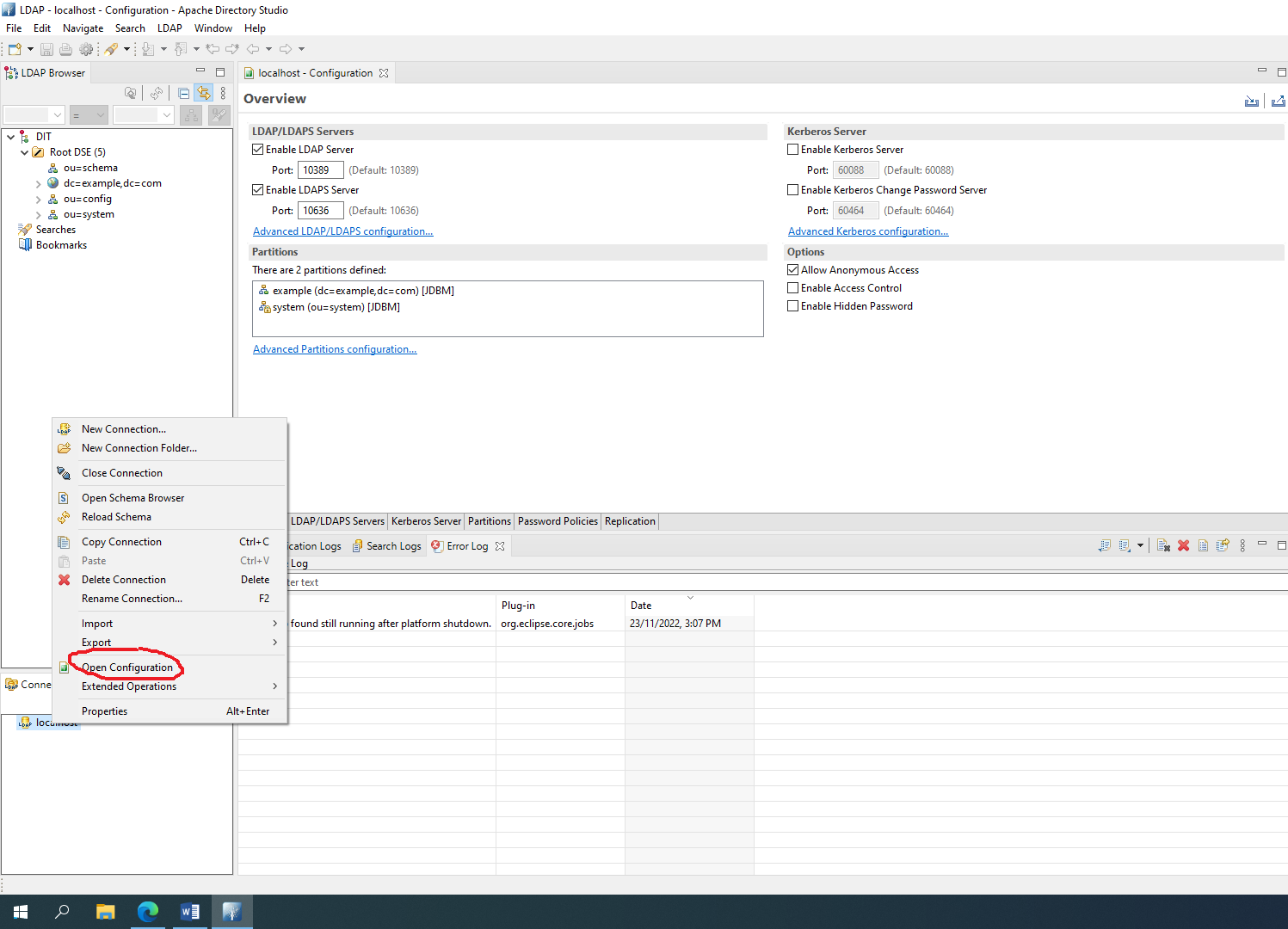
**Lưu ý**: Do các hạn chế bảo mật Unix truyền thống, các cổng nhỏ hơn 1024 là “đáng tin cậy”. Do đó trên hệ thống Unix, một quy trình không phải root phải lắng nghe trên một cổng lớn hơn 1023. Hãy nhớ, các server hiện tại đều chạy hệ điều hành linux.

Có 2 trường hợp có thể tiến hành đổi cổng:

1. Khi máy chủ chưa được khởi động.
2. Khi máy chủ được khởi động.

#### Sử dụng Apache directory studio

**Khi máy chủ được khởi động**



#### Cách thủ công

Khi máy chủ được khởi động lần đầu tiên sau khi cài đặt, ‘config.ldif’ đổi tên thành ‘config.ldif\_migrated’. Sau thời gian đó, tệp ldif khởi tạo này sẽ không bao giờ được gọi lại, vì vậy mọi thay đổi sẽ bị bỏ qua.

**Khi máy chủ chưa khởi động**

Trên Window ta vào thư mục cài đặt Apache Server, tiếp tục: ./instances/default/conf/config.ldif

Trên Linux ta vào thư mục cài đặt Apache Server, tiếp tục: ./instances/default/conf/config.ldif

Tra cứu

dn: ads-transportID=ldap 🡪 thay đổi `ads-systemport` để thay đổi port của ldap.

dn: ads-transportID=ldaps 🡪 thay đổi `ads-systemport` để thay đổi port của ldaps.

### Đổi mật khẩu quản trị.

Trong khi máy chủ đang hoạt động, ta thay đổi giá trị của thuộc tính `userPassword` của quản trị viên (uid=admin, ou=system)

### Thêm phân vùng của người dùng

#### Phân vùng là gì?

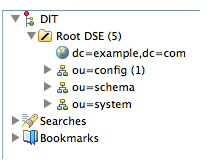
Trong ApacheDS, các entry được lưu trữ các phân vùng, mỗi phân vùng chứa 1 cây hoàn chình (DIT)

Các phân vùng không liên kết với nhau 🡪 sự thay đổi trên phân vùng A không ảnh hưởng đến phân vùng B.

Các entry trong 1 phân vùng cụ thể được lưu trữ bên dưới một số bối cảnh đặt tên được gọi là hậu tố phân vùng (partition suffix) (ví dụ o=sevenSeas trong phân vùng được tạo mới bên dưới).

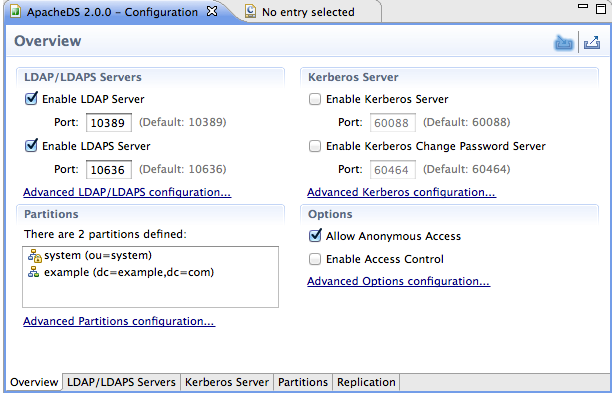
Cấu hình mặc định của ApacheDS chứa entry trong phân vùng có hậu tố “dc=example, dc=com”.

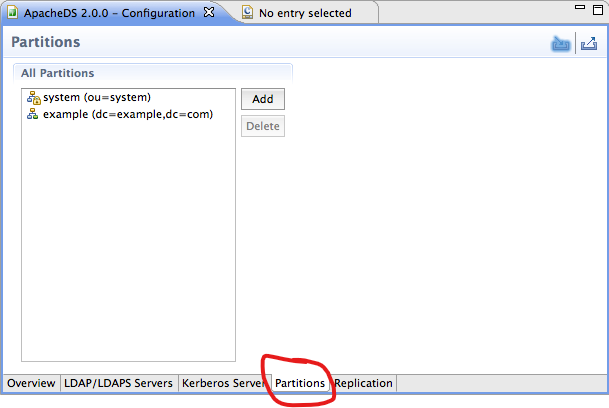
Hình ảnh bên dưới hiển thị các phân vùng mặc định của ApacheDS:

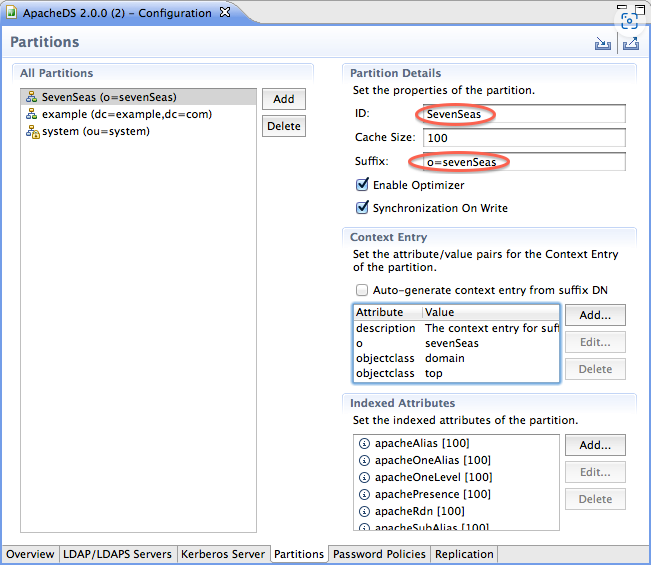


#### Định nghĩa phân vùng cơ bản

Các bước:





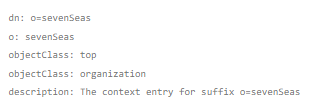


Bấm lưu bằng Ctrl + S

Sau đó khởi động lại ApacheDS server 🡪 Ta sẽ nhận được phân vùng mới là ‘SevenSeas’

**Lưu ý**: Trước khi sử dụng phân vùng trên (ví dụ: thêm entry), ta phải thêm root.

Nếu ta vẫn định tải entry bằng file .ldif vào phân vùng trên, ta chỉ cần thêm root làm entry đầu tiên:



#### Thêm 1 phân vùng bằng Java

Dưới đây là 1 đoạn mã để tạo mới phân vùng ‘=sevenSeas’ và entry của nó bằng chương trình Java:



#### Các thuộc tính cấu hình cho một phân vùng JDBM

|  |  |  |  |
| --- | --- | --- | --- |
| Thuộc tính | Mô tả | Giá trị mặc định | Bắt buộc |
| ads-partitionId | Id xác định phân vùng | N/A | Có |
| ads-partitionSuffix | Hậu tố của phân vùng | N/A | Có |
| ads-contextEntry | root | Sẽ tự động được suy ra nếu không có giá trị nào được cung cấp | Không |
| ads-optimizerEnabled |  | True | Không |
| ads-partitionCacheSize | Kích thước bộ đếm (chỉ dành cho phân vung JDBM) | -1 (no cache) | Không |
| ads-partitionSyncOnWrite | Đồng bộ hóa trên đĩa với mọi thao tác ghi | True | Không |

#### Thêm một phân vùng theo cách thủ công

Khi máy chủ được khởi động lần đầu tiên sau khi cài đặt, ‘config.ldif’ đổi tên thành ‘config.ldif\_migrated’. Sau thời gian đó, tệp ldif khởi tạo này sẽ không bao giờ được gọi lại, vì vậy mọi thay đổi sẽ bị bỏ qua.

**Trước khi khởi động máy chủ**

Ví dụ, cấu hình config.ldif để thay đổi phân vùng mặc định ‘dc=example, dc=com’ thành phân vùng của riêng bạn.

**Lưu ý**: entry của phân vùng mặc định này không có thuộc tính `ads-contextEntry`.

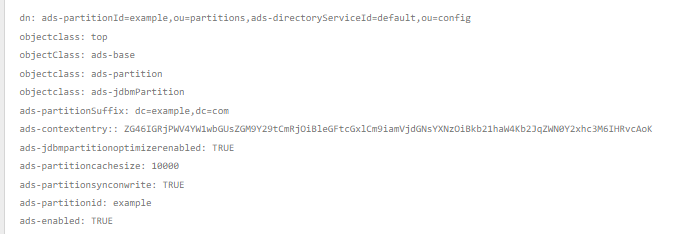
**Sử dụng ldapmodify để tạo phân vùng khi một máy chủ đang chạy**

Các phân vùng mới này bắt buộc phải có `ads-contextEntry` và giá trị của nó là:



được chuyển sang mảng byte[] mã hóa Base64

Ví dụ phân vùng ‘dc=example, dc=com’:



Ngoài ra bất kỳ thuộc tính tính utf-8 nào cũng phải được mã hóa nhị phận 🡪 base64.

Ví dụ: dn : cn=lécharyny, dc =com 🡪 cần được định nghĩa: dn: cn=Y249bMOpY2hhcm55LGRjWNvbQ==

### Định nghĩa ghi log

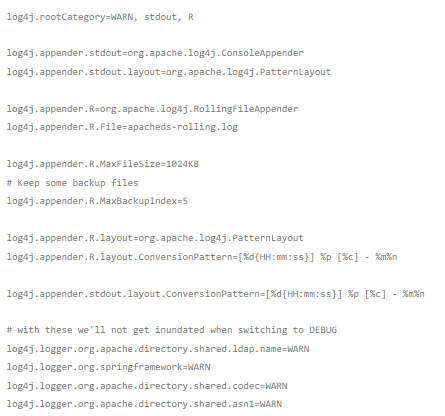
#### ApacheDS và ghi nhật ký

ApacheDS 2.0 sử dụng Slf4j làm giải pháp ghi nhật ký. Giá trị mặc định cho ApacheDS 2.0 là log4j

#### Hành vi mặc định sau khi cài đặt

Theo mặc định, ApacheDS ghi các tệp log trong thư mục <APACHEDS\_HOME>/instances/default/conf/long4j.properties

Bên cạnh thiết bị xuất chuẩn, một `RollingFileAppender` được sử dụng để thu thập các warn và error. Tên của `RollingFileAppender` là ‘R’;

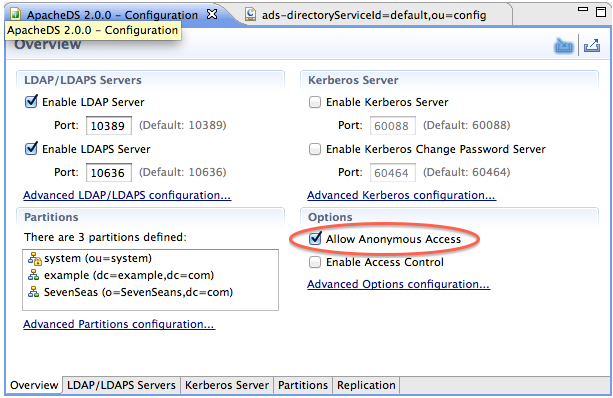


🡺Nói chung là giống java.

### Bật và tắt truy cập ẩn danh

Truy cập ẩn danh vào máy chủ được bật theo mặc định. Điều này bao gồm đọc và ghi.

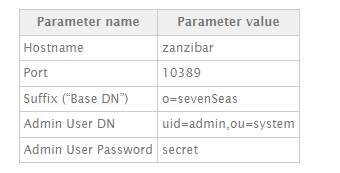
Ta có thể tắt các liên kết ẩn danh bằng cách thay đổi giá trị thuộc tính `Allow Anonymous Access` trong màn hình:



Việc khởi động lại máy chủ là cần thiết để thay đổi có hiệu lực. Sau đó, tất cả người dùng phải cung cấp tên của họ (distinguished name) và mật khẩu để liên kết với LDAP.

### Cấu hình mẫu và dữ liệu thư mục mẫu

#### Thông số máy chủ cơ bản



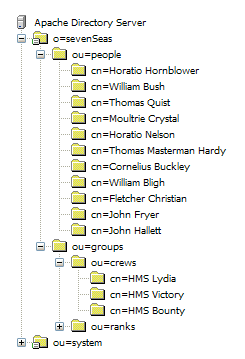
#### LDAP client

LDAP là một giao thức client/server. Do đó, ta cần một ứng dụng khách LDAP để kết nối từ xa với Máy chủ thư mục Apache.

Có nhiều lựa chọn khác nhau ở đây. Vì giao thức được chuẩn hóa nên ta có thể sử dụng mọi ứng dụng khách tuân thủ LDAP. Điều này có thể so sánh với HTTP, nơi bạn có thể dụng mọi trình duyệt web để giao tiếp với hầu như mọi máy chủ web và hoàn toán khác với cơ sơ dữ liệu quan hệ.

Trong mọi trường hợp, ví dự trên sẽ được kết nối với máy chủ LDAP bằng đường dẫn: ldap://zanzibar:10389/o=sevenSeas

#### Dữ liệu mẫu



Không có file đâu, document của ApacheDS không có.

### Phục hồi dữ liệu đã lưu

ApacheDS lưu trữ dữ liệu của nó vào một tập hợp các tệp nhị phân 🡪 vì vậy ta không thể đơn giản sao chép các tệp đó vào thư mục sao lưu.

Ta chỉ có thể trích xuất file .ldif từ ApacheDS.

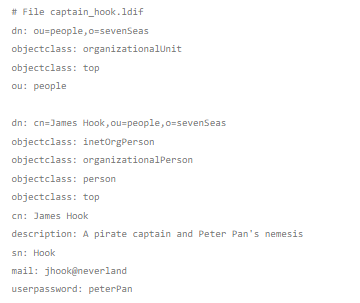
Ví dụ, đây là một dòng lệnh mẫu trích xuất dữ liệu từ phân vùng `dc=example, dc=com`:



## Thêm, xóa, sửa dữ liệu

### Thêm entry

Ví dụ một entry sẽ được thêm vào phân vùng ‘SevenSeas’:



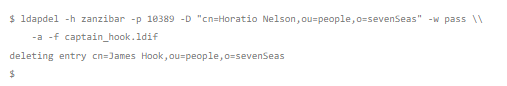
Entry đầu tiên tạo vùng chứa ou=people, điều này là cần thiết, nếu không ta không thể thêm entry thứ 2.

**Sử dụng dòng lệnh**

Sử dụng người dùng được ủy quyền ‘cn=Horatio Nelson, ou=people, o=sevenSeas’

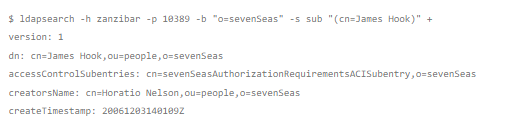


### Xóa entry



### Tìm kiếm đơn giản

Ta có thể sử dụng dấu ‘+’ để nhận thông tin khi thêm entry thành công:



## Bảo mật cơ bản

### Tùy chọn xác thực

#### Xác thực là gì?

Trong ApacheDS, ta có thể muốn xác định máy khách để kiểm tra xem chúng có được phép đọc, thêm hoặc thao tác một số dữ liệu được lưu trữ trong thư mục hay không. Cũng như một người dùng được xác thực có được phép làm điều gì hay không.

Khá thường xuyên, quá trình xác thực được ủy quyền cho một dịch vụ thư mục (ví dụ: ApacheDS, …) bởi các phần mềm khác. Bởi vì khi làm như vậy, dữ liệu xác thực (ví dụ: tên người dùng, mật khẩu) và dữ liệu ủy quyền (ví dụ: mối quan hệ nhóm) được lưu trữ và quản lý tập trung.

ApacheDS 2.0 hỗ trợ xác thực đơn giản và liên kết ẩn danh, trong khi lưu trữ mật khẩu trong thuộc tính `userPassword` tại entry của người dùng.

#### Ràng buộc đơn giản

Phương pháp ràng buộc đơn giản được ApacheDS 2.0 hỗ trợ cho tất cả entry người dùng lưu trữ ở bất kỳ phân vùng nào, nếu chúng chứa mật khẩu.

Quy trình:

* LDAP client cung cấp `dn` người dùng và mật khẩu cho máy chủ LDAP. ApacheDS kiểm tra xem mật khẩu đó có giống với mật khẩu được lưu trữ trong thuộc tính `userPassword` của entry client đã cung cấp hay không?

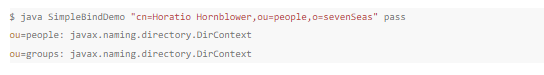
#### Sử dụng công cụ dòng lệnh



#### Liên kết từ các thành phần JAVA sử dụng JNDI



Nếu DN của entry người dùng và mật khẩu phù hợp thì chương trình sẽ liên kết thành công và thực hiện tìm kiếm



Nếu không thì:



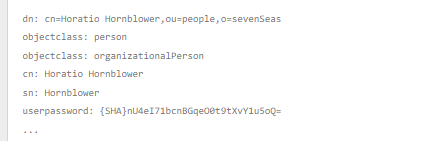
Trong cuộc sống, ta thường muốn tách hầu hết dữ liệu cấu hình khỏi mã nguồn, chẳng hạn như với sự giúp đỡ của tệp jindi.properties

#### Mật khẩu lưu trữ được mã hóa một chiều

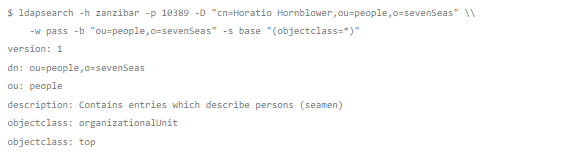
Nếu mật khẩu được lưu trữ trong thư mục rõ ràng như trên, quản trị viên (uid=admin, ou=system) có thể đọc chúng. Điều này đúng ngay cả khi ủy quyền được bật. Mật khẩu cũng sẽ hiển thị trong các tệp LDIF đã xuất.

**Lưu ý**: Nếu một người không sử dụng SSL, mật khẩu sẽ được truyền dưới dạng văn bản rõ ràng.

**Mật khẩu khoog được lưu trữ dưới dạng văn bản rõ ràng**

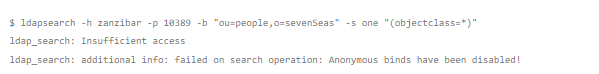


**Xác thực**

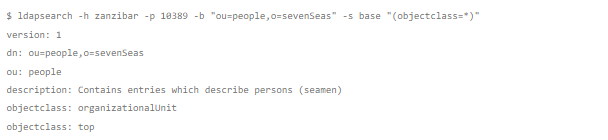
****

#### Liên kết ẩn danh

Ví dụ hành vi của máy chủ với các liên kết ẩn danh bị vô hiệu hóa

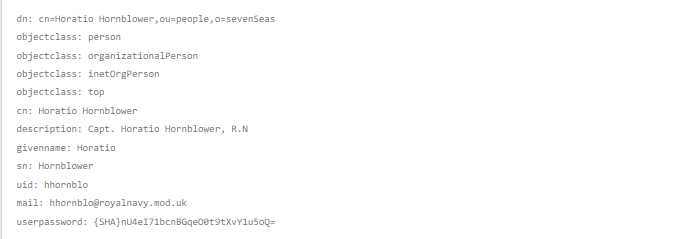


Ví dụ: hành vi của máy chủ với các liên kết ẩn danh được bật



#### Xác thực người dùng bằng uid và mật khẩu

Ví dụ entry của người dùng:



**Thuật toán**

Đầu vào:

* uid của người dung và mật khẩu

Các bước:

1. Liên kết với ApacheDS bằng liên kết ẩn danh hoặc `dn` của người quản lý.
2. Thực hiện thao tác tìm kiếm với bộ lọc thích hợp để tìm entry của người dùng với ID đã cho. Trong ví dụ là ‘(&(objectClass=inetorgperson)(uid=hhornblo))’
3. Liên kết với ApacheDS bằng `dn` vừa tìm được và mật khẩu được cung cấp làm đối số

**Mã mẫu**

**import** **java.util.Hashtable**;

**import** **javax.naming.Context**;

**import** **javax.naming.NamingEnumeration**;

**import** **javax.naming.NamingException**;

**import** **javax.naming.directory.DirContext**;

**import** **javax.naming.directory.InitialDirContext**;

**import** **javax.naming.directory.SearchControls**;

**import** **javax.naming.directory.SearchResult**;

**public** **class** AdvancedBindDemo {

**public** **static** **void** main(String[] args) **throws** NamingException {

**if** (args.length < 2) {

System.err.println("Usage: java AdvancedBindDemo <uid> <password>");

System.exit(1);

}

Hashtable env = **new** Hashtable();

env.put(Context.INITIAL\_CONTEXT\_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");

env.put(Context.PROVIDER\_URL, "ldap://zanzibar:10389/");

env.put(Context.SECURITY\_AUTHENTICATION, "simple");

String uid = args[0];

String password = args[1];

DirContext ctx = **null**;

**try** {

*// Step 1: Bind anonymously*

ctx = **new** InitialDirContext(env);

*// Step 2: Search the directory*

String base = "o=sevenSeas";

String filter = "(&(objectClass=inetOrgPerson)(uid={0}))";

SearchControls ctls = **new** SearchControls();

ctls.setSearchScope(SearchControls.SUBTREE\_SCOPE);

ctls.setReturningAttributes(**new** String[0]);

ctls.setReturningObjFlag(**true**);

NamingEnumeration enm = ctx.search(base, filter, **new** String[] { uid }, ctls);

String dn = **null**;

**if** (enm.hasMore()) {

SearchResult result = (SearchResult) enm.next();

dn = result.getNameInNamespace();

System.out.println("dn: "+dn);

}

**if** (dn == **null** || enm.hasMore()) {

*// uid not found or not unique*

**throw** **new** NamingException("Authentication failed");

}

*// Step 3: Bind with found DN and given password*

ctx.addToEnvironment(Context.SECURITY\_PRINCIPAL, dn);

ctx.addToEnvironment(Context.SECURITY\_CREDENTIALS, password);

*// Perform a lookup in order to force a bind operation with JNDI*

ctx.lookup(dn);

System.out.println("Authentication successful");

enm.close();

} **catch** (NamingException e) {

System.out.println(e.getMessage());

} **finally** {

ctx.close();

}

}

}

### Phân quyền cơ bản

#### Ủy quyền là gì?

Ủy quyền cho các hoạt động thư mục không được chuẩn hóa nghiêm ngặt trong thế giới LDAP, bởi RFC 2829 – Phương pháp xác thực cho LDAP mô tả kich bản và khái niệm khác nhau 🡪 mỗi máy chủ LDAP cung cấp một tính năng ủy quyền riêng.

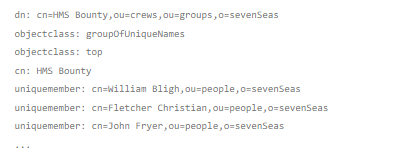
ApacheDS cung cấp hệ thống con ủy quyền mạnh mẽ từ phiên bản 0.9.3, nhưng bị tắt theo mặc định.

#### Ủy quyền cho các hoạt động thư mục so với thành viên nhóm

Để thực hiện ủy quyền của mình, các thành phần phần mềm thuồng tận dụng các nhóm LDAP có sẵn.

* ‘groupOfNames’ và ‘groupOfUniqueNames’ là các objectclass phổ biến cho các group entry, chúng chứa các `dn` của thành viên trong nhóm.

Ví dụ:

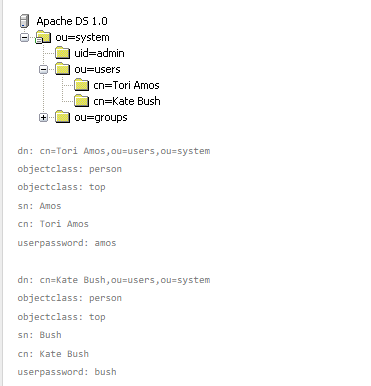


#### Hành vi ủy quyền mặc định cho các hoạt động thư mục

Nếu không bật kiểm soát truy cập, tất cả các entry đều có thể bị truy cập và thay đổi bởi tất cả người dùng, kể cả người dùng ẩn danh.

Tuy nhiên, có một số quy tắc tích hợp để tối thiểu bảo vệ người dùng và nhóm trong máy chủ mà không cần phải bật hệ thống con ACI.

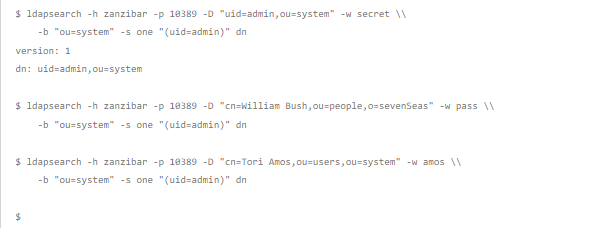
**Dữ liệu mẫu trong ‘ou=users, ou=system’**



Chúng được sử dụng trong các ví dụ bên dưới, kết hợp với phân vùng ‘o=sevenSeas’, để mô tả các quy tắc uy quyền mặc định

#### Quy tắc và hoạt động mẫu

Nếu không có ACI, máy chủ sẽ tự động bảo vệ, ẩn người dùng quản trị khỏi mọi người, trừ người quản trị. Dưới đây là ví dụ:



Người dùng không thể thấy các entry của người khác trong entry ‘ou=users, ou=system’. Vì vậy, việc đặt người dùng mới ở đó sẽ tự động bảo vệ họ.

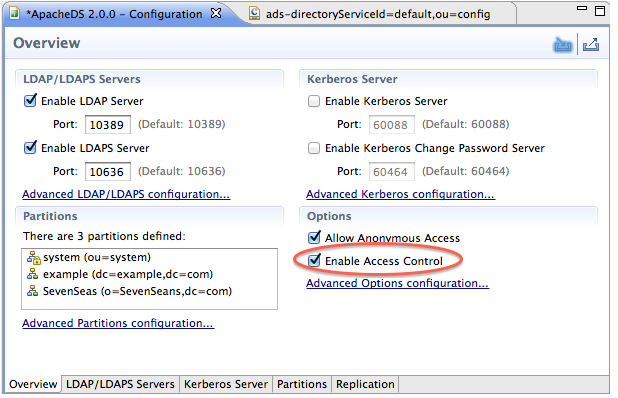


### Ví dụ đơn giản cho hệ thống con ACI

#### Yêu cầu

* Phân vùng ‘o=sevenSeas’ được sử dụng để làm phân vùng kiếm soát truy cập.
* Người dùng ‘cn=Horatio Nelsonm, ou=people, o=sevenSeas’ sẽ có thể thực hiện tất cả các thao tác (thêm, xóa, sửa, …) tại phân vùng ‘o=sevenSeas’
* Tất cả người dùng khác và người dùng ẩn danh chỉ có thể tìm kiếm và so sánh (không thể thêm, xóa, sửa, …)
* Tất cả người dùng khác và người dùng ẩn danh không thể đọc thuộc tính `userPassword`

#### Kích hoạt hệ thống con ACI



#### Các tác vụ cấu hình khác để thực hiện sau đó

1. Tạo một thuộc tính `administratorRole` với giá trị ‘accessControlSpecificArea’ trong phân vùng ‘o=sevenSeas’
2. Tạo root cho ‘o=sevenSeas’ để cấp quyền cho tất cả các hoạt động của ‘cn=Horatio Nelson, ou=people, o=sevenSeas’, người đóng vai trò quản lý thư mục.

Root phải chứa các thuộc tính và giá trị sau:



1. Một giá trị thuộc tính mới sẽ được thêm vào root để cấp quyền tìm kiếm cho tất cả người dùng:



1. Một giá trị thuộc tính mới sẽ được thêm vào root để quy định từ chối tìm kiếm và so sánh `userPassword` đối với tất cả người dùng



🡺Thuộc tính 3 và 4 có thể kết hợp vào 1 thuộc tính duy nhất:



#### File .ldif cấu hình

# File authz\_sevenSeas.ldif

#

# Create an operational attribute "administrativeRole"

# with value "accessControlSpecificArea" in the entry "o=sevenSeas".

#

dn: o=sevenSeas

changetype: modify

add: administrativeRole

administrativeRole: accessControlSpecificArea

# Create a subentry subordinate to "o=sevenSeas" to grant all operations' permissions

# to "cn=Horatio Nelson,ou=people,o=sevenSeas", to grant search and compare permissions

# to all users and to deny search and compare permissions for userPassword attribute to all users.

#

dn: cn=sevenSeasAuthorizationRequirementsACISubentry,o=sevenSeas

changetype: add

objectclass: top

objectclass: subentry

objectclass: accessControlSubentry

cn: sevenSeasAuthorizationRequirementsACISubentry

subtreeSpecification: {}

prescriptiveACI: {

identificationTag "directoryManagerFullAccessACI",

precedence 11,

authenticationLevel simple,

itemOrUserFirst userFirst:

{

userClasses

{

name { "cn=Horatio Nelson,ou=people,o=sevenSeas" }

},

userPermissions

{

{

protectedItems

{

entry, allUserAttributeTypesAndValues

},

grantsAndDenials

{

grantAdd, grantDiscloseOnError, grantRead,

grantRemove, grantBrowse, grantExport, grantImport,

grantModify, grantRename, grantReturnDN,

grantCompare, grantFilterMatch, grantInvoke

}

}

}

}

}

prescriptiveACI: {

identificationTag "allUsersACI",

precedence 10,

authenticationLevel none,

itemOrUserFirst userFirst:

{

userClasses

{

allUsers

},

userPermissions

{

{

protectedItems { entry, allUserAttributeTypesAndValues },

grantsAndDenials { grantRead, grantBrowse, grantReturnDN,

grantCompare, grantFilterMatch, grantDiscloseOnError }

},

{

protectedItems { attributeType { userPassword } },

grantsAndDenials { denyRead, denyCompare, denyFilterMatch }

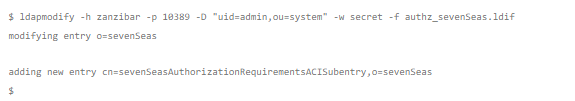
}

}

}

}

Để áp dụng cấu hình này, ta sử dụng lệnh sau:



Tiếp tục đọc tiếp tại: [3.3 - How to enable SSL — Apache Directory](https://directory.apache.org/apacheds/basic-ug/3.3-enabling-ssl.html)