

TRƯỜNG ĐẠI HỌC CÀN THƠ
KHOA CÔNG NGHỆ THÔNG TIN – TRUYỀN THÔNG
BỘ MÔN HỆ THỐNG MÁY TÍNH – TRUYỀN THÔNG



Luận Văn Tốt Nghiệp

ĐỀ TÀI:

*Cài Đặt Cơ Chế Chứng Thực Một Cửa SSO
(Single-Sign-On) Trên Liferay Portal*

GIÁO VIÊN RA ĐỀ TÀI:

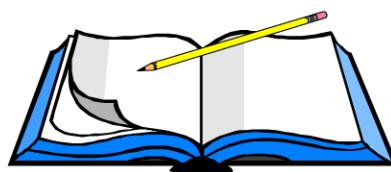
Giảng Viên: Tiến Sĩ Ngô Bá Hùng
MSCB: 1124

Email: [nbhunger@cit.ctu.edu.vn](mailto:nbhung@cit.ctu.edu.vn)

Bộ môn: Hệ Thống Máy Tính – Truyền Thông

SINH VIÊN THỰC HIỆN

Dương Thế Anh
MSSV: 1071656
Lớp: DI0797A1 – K33



Cần Thơ 05/2011

-= * Học kỳ 2, niên khóa 2010-2011 *=-

LỜI CẢM ƠN

Đầu tiên tôi xin gửi lời cảm ơn chân thành đến thầy, Tiến sĩ **Ngô Bá Hùng**, người đã hướng dẫn và giúp đỡ tôi trong suốt quá trình thực hiện luận văn.

Tôi cũng xin gửi lời cảm ơn đến tất cả thầy cô khoa **Công nghệ thông tin & Truyền Thông, Trường Đại học Cần Thơ**. Các thầy cô đã tận tình giảng dạy và truyền đạt những kiến thức bổ ích cho tôi trong suốt những năm học qua, giúp tôi có được những kiến thức cần thiết cho việc thực hiện bài luận văn cũng như những tri thức làm hành trang cho tôi vào đời.

Tôi gửi lời cảm ơn đến những người bạn của tôi, các bạn đã giúp đỡ tôi rất nhiều để tôi có thể hoàn thành bài luận của mình. Tôi xin cảm ơn vì các bạn đã động viên giúp đỡ tôi trong lúc tôi gặp khó khăn nhất.

Cuối cùng tôi xin tỏ lòng biết ơn chân thành nhất đến ba mẹ, ba mẹ đã tạo mọi điều kiện tốt nhất để tôi có thể học tập, khuyến khích động viên khi tôi cảm thấy chán nản và muốn bỏ cuộc.

Cần Thơ, ngày tháng 04 năm 2010

Sinh viên thực hiện

Dương Thé Anh (MSSV 1071656)

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN



Cần Thơ, Ngày..... Tháng..... Năm 2011

Giáo viên hướng dẫn

TS. Ngô Bá Hùng

NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN



Cần Thơ, Ngày..... Tháng..... Năm 2011

Giảng Viên Phản Biện

Mục Lục

LỜI CẢM ƠN.....	1
Mục Lục	4
Tóm Tắt	8
Chương 1: Tổng quan	9
1.1 Đặt vấn đề:.....	9
1.2 Mục tiêu cần đạt được:	9
1.3 Hướng giải quyết:	9
Chương 2: Cơ sở lý thuyết	11
2.1 Tìm hiểu Single Sign-On	11
2.1.1 Khái niệm	11
2.1.2 Lợi ích:.....	11
2.1.3 Các giải pháp Single Sign-On:	12
2.1.4 Central Authenticate Service (CAS)	12
2.2 Lightweight Directory Access Protocol:	34
2.2.1 Khái niệm:	34
2.2.2 Phương thức hoạt động của LDAP:.....	34
2.2.3 Các thao tác trên giao thức LDAP:.....	37
2.2.4 Tính năng mở rộng của LDAP:	39
2.2.5 Các mô hình LDAP:	40
2.2.6 Trình xử lý xác thực LDAP – CAS:	43
2.3 Giới thiệu Liferay Portal:.....	45
2.3.1 Portal là gì?.....	45
2.3.2 Thuận lợi của Portal.....	46
2.3.3 Tạo một portal với Liferay:	47
2.3.4 Cấu trúc bên trong trang:	51
2.3.5 Tiến trình tạo trang:	52
2.3.6 Cách giải quyết yêu cầu:.....	53
2.3.7 Quản lý nội dung với CMS:.....	54

2.4 Moodle:.....	55
2.4.1 Moodle là gì?	55
2.4.2 Tại sao dùng Moodle?	57
2.4.3 Sử dụng Moodle:	59
2.5 Servlet:.....	59
2.5.1 Servlet là gì?	59
2.5.2 Kiến trúc Servlet cơ bản:	60
Chương 3: Ứng dụng thực tiễn	62
3.1 Xây dựng cổng thông tin CICT:	62
3.1.1 Đối tượng người dùng trong hệ thống CICT Portal	62
3.1.2 Cài đặt các trang trong hệ thống	63
3.2 Triển khai Central Authentication Service (CAS):	64
3.2.1 Triển khai cơ chế xác thực CAS – LDAP:	67
3.2.2 Cấu hình Liferay sử dụng CAS và LDAP:	70
3.2.3 Triển khai CAS với Moodle:	75
3.2.4 Triển khai CAS với Servlet:	78
Chương 4: Kết luận.....	80
4.1 Kết quả đạt được.....	80
4.2 Hạn chế	80
4.3 Hướng phát triển:.....	80
Phụ lục	81
Tài Liệu Tham Khảo.....	81
Hướng dẫn cài đặt LDAP Server trên Ubuntu 10.10	81

DANH MỤC HÌNH ẢNH

Hình 1. 1 Trước và sau khi sử dụng Single Sign-On.....	11
Hình 1. 2 Người dùng truy cập vào ứng dụng mà chưa chứng thực với CAS server	14
Hình 1. 3 Người dùng truy cập vào ứng dụng khi đã chứng thực với CAS server	15
Hình 1. 4 CAS - luồng đăng nhập (1).....	29
Hình 1. 5 CAS - luồng đăng nhập (2).....	30
Hình 1. 6 CAS - luồng đăng nhập (3).....	31
Hình 1. 7 CAS - luồng đăng nhập (4).....	32
Hình 1. 8 CAS - Proxy Flow	33
Hình 1. 9 CAS – Luồng đăng xuất	33
Hình 2. 1 Một client nhận một entry từ directory	35
Hình 2. 2 Một client nhận nhiều entry từ directory	36
Hình 2. 3 Một client phát ra nhiều yêu cầu tìm kiếm đồng thời.....	36
Hình 2. 4 Một dạng trao đổi LDAP	37
Hình 2. 5 Một ứng dụng Directory-Enabled thực hiện một công việc phức tạp.....	38
Hình 2. 6 Một ứng dụng Directory-Enabled Server	39
Hình 2. 7 Một cây thư mục LDAP	40
Hình 2. 8 Hệ thống tập tin trong UNIX	41
Hình 2. 9 Một phần thư mục LDAP với các entry chứa thông tin	42
Hình 3. 1 Một portal sử dụng Liferay	49
Hình 3. 2 Cấu trúc bên trong một trang portal	51
Hình 3. 3 Một trang portal được sinh ra như thế nào	52
Hình 3. 4 Sơ đồ trình tự về yêu cầu người dùng	53
Hình 3. 5 Vòng đời Servlet đặc trưng.....	60
Hình 4. 1 Sơ đồ khái quát về hệ thống người dùng trong PQID Portal.....	62
Hình 4. 2 Cấu Trúc Trang	63
Hình 4. 3 Mô hình triển khai CAS Server – Liferay - Moodle	65
Hình 4. 4 Tạo keystore	66
Hình 4. 5 Xuất ra tập tin tomcatssl.cert	66
Hình 4. 6 Import tập tin tomcatssl.cert vào keystore của JAVA	67
Hình 4. 7 Cấu Hình server.xml của Tomcat	67
Hình 4. 8 Tập tin pom.xml	68

Hình 4. 9	Chỉnh sửa tập tin deployerConfigContext.xml.....	68
Hình 4. 10	Thiết lập biến môi trường path cho maven.....	69
Hình 4. 11	Build CAS Server qua Maven đã thành công.....	69
Hình 4. 12	Thư mục target.....	70
Hình 4. 13	Giao diện cấu hình Authentication trong liferay	70
Hình 4. 14	Cấu hình Liferay - LDAP	71
Hình 4. 15	Tên Server.....	71
Hình 4. 16	Thiết lập các thông số connection đến LDAP Server.....	71
Hình 4. 17	Thiết lập các thông số về LDAP User	72
Hình 4. 18	Thiết lập các tham số về Group trong LDAP	73
Hình 4. 19	Một vài thuộc tính đồng bộ từ Liferay đến LDAP	73
Hình 4. 20	Đồng bộ mỗi khi khởi động Liferay	74
Hình 4. 21	Thiết lập CAS với Liferay	74
Hình 4. 22	Import certificate vào JRE của Liferay	75
Hình 4. 23	Giao diện Manage Authentication.....	75
Hình 4. 24	Một số tham số về cấu hình CAS Server.....	76
Hình 4. 25	Thiết lập LDAP Server	76
Hình 4. 26	Kết nối đến LDAP	76
Hình 4. 27	Thiết lập về người dùng LDAP	77
Hình 4. 28	Data Mapping	77
Hình 4. 29	Thiết lập CAS và Servlet	79

Tóm Tắt

Ngày nay, nhu cầu tìm kiếm thông tin từ Internet ngày càng nhiều. Cổng thông tin là một trong những nguồn cung cấp thông tin đang được áp dụng rộng rãi trên toàn thế giới. Khuynh hướng các dịch vụ cùng nhau chia sẻ dữ liệu người dùng đang là hướng phát triển chung của công nghệ thông tin, cùng với đó là nhu cầu đăng nhập một lần cho các dịch vụ. Đề tài “CÀI ĐẶT CƠ CHẾ CHỨNG THỰC MỘT CỬA SSO (SINGLE SIGN-ON) TRÊN LIFERAY” được đưa ra nghiên cứu nhằm giới thiệu về một cổng thông tin mã nguồn mở Liferay, và hướng dẫn cài đặt cơ chế đăng nhập một lần trên cổng thông tin đáp ứng nhu cầu trên.

Trong quá trình nghiên cứu, tôi đã sử dụng Liferay Portal Community Edition phiên bản 6.0.5, JASIG CAS (Central Authentication Service) phiên bản 3.4.4. Cùng với đó là chạy một máy ảo UBUNTU 10.10 đóng vai trò là một LDAP (Lightweight Directory Access Protocol) để quản lý người dùng tập trung, cùng với một số dịch vụ khác như Moodle, Servlet. Nhìn chung, đã đáp ứng được mục tiêu của đề tài. Bên cạnh đó vẫn có một số vấn đề chưa được giải quyết như: giao diện chưa được chú tâm nhiều, phân quyền trong từng hệ thống còn lỏng lẻo. Tuy nhiên, cổng thông tin vẫn có khả năng tiếp tục phát triển và triển khai trong thực tiễn.

Chương 1: Tổng quan

1.1 Đặt vấn đề:

Làm thế nào giải quyết được cơ chế chứng thực một cửa (Single Sign-On) để tích hợp portal, các ứng dụng web.

Với một nền kinh tế hội nhập, nhu cầu cập nhật tin tức từ nhiều nguồn khác nhau là rất cần thiết. Ngoài sách, báo, tạp chí, truyền hình, thì tin tức qua Internet là một nguồn cung cấp thông tin không kém phần quan trọng và nhanh, mà cụ thể là các cổng thông tin (portal) là một trong những nguồn cung cấp thông tin trên Internet được áp dụng rộng rãi trên toàn thế giới. Hiện nay, số lượng cổng thông tin mã nguồn mở cũng tương đối nhiều, nhưng để đánh giá cái nào tốt hơn thì rất khó, các nhà phát triển lựa chọn tùy theo tính dễ dàng phát triển, sự giàu có các chức năng, sự tùy biến trong giao diện, và kiến trúc có khả năng gắn nối, thêm vào đó là sự phổ biến của portal để có sự hỗ trợ tốt của cộng đồng thì Liferay Portal là một sự lựa chọn tốt.

Khi đã xây dựng được một portal, thì việc phát triển những ứng dụng web, dịch vụ web tích hợp vào trong portal là rất cần. Vấn đề xảy ra ở đây, với một tài khoản người dùng, đăng nhập vào một ứng dụng hay dịch vụ, nhưng khi người dùng chuyển qua một dịch vụ khác trong cùng hệ thống thì sẽ không phải bị nhắc nhớ đăng nhập lại, mà sẽ sử dụng quyền truy xuất qua lần đăng nhập đầu tiên để vào. Và đó cũng chính là tác dụng của cơ chế chứng thực một cửa khi tích hợp vào portal và các ứng dụng web.

1.2 Mục tiêu cần đạt được:

Trước vấn đề đặt ra như thế, có một số mục tiêu cần đạt được:

- Nghiên cứu cơ chế Single Sign-On (SSO).
- Tìm hiểu Liferay Portal, nghiên cứu vấn đề tích hợp cơ chế SSO vào Liferay Portal.
- Nghiên cứu tích hợp các ứng dụng mã nguồn mở sẵn có (ví dụ: Moodle,...) hỗ trợ SSO vào Liferay Portal với cơ chế SSO.
- Nghiên cứu tích hợp các ứng dụng tự phát triển (ví dụ: Servlet,...) vào Liferay Portal với cơ chế SSO.

1.3 Hướng giải quyết:

Để giải quyết những mục tiêu đã nêu được định hướng bài toán như sau:

- Sử dụng Central Authentication Service (CAS) và LDAP Server quản lý tài khoản người dùng tập trung để thực hiện cơ chế SSO.

- Tìm hiểu và xây dựng cổng thông tin Khoa Công Nghệ Thông Tin Trường Đại Học Cần Thơ (CICT Portal) dựa trên Liferay Portal.
- Cài đặt Liferay Portal với cơ chế chứng thực thực CAS, LDAP.
- Tìm hiểu và xây dựng Trang học tập trực tuyến dựa trên Moodle, tích hợp vào CICT Portal với cơ chế chứng thực CAS.
- Tạo một ứng dụng Servlet tích hợp vào hệ thống CICT Portal với cơ chế chứng thực CAS.

Chương 2: Cơ sở lý thuyết

2.1 Tìm hiểu Single Sign-On

2.1.1 Khái niệm

Single Sign-On (SSO) là một đặc tính cho phép người dùng chỉ phải đăng nhập một lần và truy xuất đến tất cả các dịch vụ trong hệ thống mà không bị nhắc nhở đăng nhập lại tại mỗi hệ thống.

Ví dụ:



Hình 1.1 Trước và sau khi sử dụng Single Sign-On

Người dùng sử dụng nhiều dịch vụ như: Đăng ký môn học, hệ thống xem điểm, ... ứng mỗi dịch vụ chúng ta có một tài khoản riêng. Trước đây, khi chưa sử dụng SSO thì khi với mỗi dịch vụ người dùng đều phải nhập thông tin để xác thực. Khi một tổ chức đã thống nhất sử dụng SSO cho tất cả các dịch vụ của họ thì người dùng chỉ cần đăng nhập một lần duy nhất trên bất kỳ dịch vụ nào trong tổ chức, do đó khi truy xuất những dịch vụ khác, người dùng không cần phải đăng nhập lại.

2.1.2 Lợi ích:

- ✓ Tránh việc nhớ nhiều thông tin đăng nhập (username, password) khi dùng nhiều dịch vụ.
- ✓ Tiết kiệm thời gian khi tái lập lại mật khẩu cho một người dùng (identity user).
- ✓ Bảo mật tất cả các cấp độ của việc thoát hay truy xuất vào hệ thống.

- ✓ Người phát triển ứng dụng không cần phải hiểu và thực hiện nhận dạng bảo mật trong ứng dụng của họ.

2.1.3 Các giải pháp Single Sign-On:

- Open Single Sign-On (OpenSSO) hoạt động dựa trên Token.
- Central Authenticate Service (CAS) hoạt động dựa trên Ticket.
- Java Open Single Sign-On (JOSSO).

Ở đề tài này, chúng ta chỉ chú ý đến giải pháp CAS.

2.1.4 Central Authenticate Service (CAS)

2.1.4.1 Tổng quan:

JA-SIG Central Authentication Service là một enterprise level, mã nguồn mở, là giải pháp Single Sign-On với một thành phần máy chủ JAVA và nhiều thư viện hỗ trợ các client khác nhau được viết với nhiều ngôn ngữ như PHP, PL/SQL, JAVA,... CAS là giao thức dựa trên HTTP, yêu cầu mỗi thành phần của CAS khi truy xuất đều phải thông qua những Uniform Resource Identifier (URI) cụ thể. CAS được phát triển lần đầu tiên bởi Trường Đại Học Yale cho giải pháp SSO.

CAS đảm nhận tính năng SSO thông qua Cookie. Cookie sẽ bị hủy khi người dùng đăng xuất khỏi CAS hay khi đóng trình duyệt. Cookie được sinh ra bởi CAS được gọi là TGC (Ticket Granting Cookie), TGC chứa một chỉ danh (ID) duy nhất cùng với thời gian hết hạn (expiration time) của cookie. Expiration time thường là 8 giờ.

CAS hỗ trợ nhiều trình xử lý xác thực (authentication handler) để xác thực thông tin. Các nhà phát triển (Developer) cũng có thể sử dụng trình xử lý xác thực riêng do họ phát triển. Những thông tin được xác thực bởi CAS chẳng hạn như username/password, X509 certificates,... Để xác thực những loại thông tin khác nhau thì CAS sử dụng các trình xử lý xác thực khác nhau.

CAS còn hỗ trợ tính năng “Remember Me”. Tính năng “Remember Me” đòi hỏi phải được cấu hình trên nhiều tập tin khác nhau và khi người dùng chọn tính năng này trên giao diện đăng nhập thì thông tin người dùng sẽ được ghi nhớ trong một khoảng thời gian được cấu hình (mặc định là 3 tháng). Và khi người dùng này quay trở lại sử dụng dịch thì sẽ chuyển đến dịch url dịch vụ tương ứng mà không phải qua giao diện đăng nhập ngay cả khi người dùng mở một cửa sổ trình duyệt mới.

2.1.4.2 Các phiên bản của CAS:

➤ CAS 1.0

- Được tạo bởi Yale University, khởi đầu từ năm 1999.
- Là một Web Single Sign On, dễ sử dụng.

➤ CAS 2.0

- Cũng được tạo ra bởi Yale University.
- Giới thiệu thêm tính năng mới là Proxy Authentication.

➤ YA-SIG CAS 3.0

- Trở thành JA-SIG project vào 2004.
- Mục đích là làm cho CAS tương thích cao hơn, mềm dẻo hơn.
- Hoàn toàn tương thích với CAS 2.0.

2.1.4.3 Nguyên tắc hoạt động của CAS:

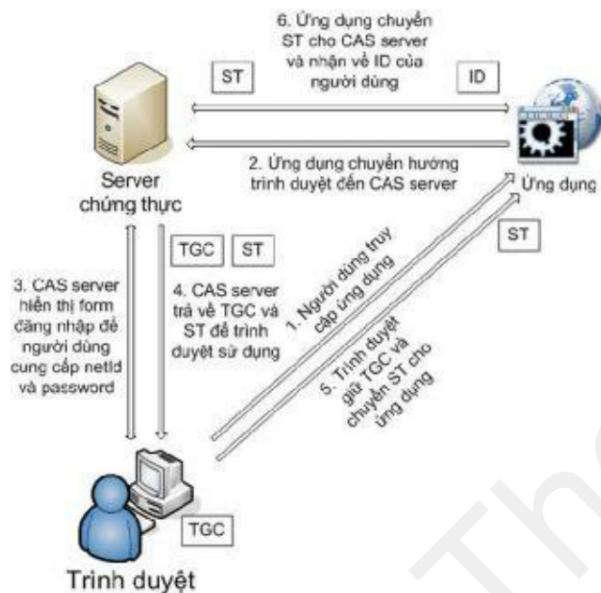
✚ **Chứng thực người dùng với CAS Server:**

Người dùng nhập tên đăng nhập (username) và mật khẩu (password) vào khung đăng nhập. Các thông tin này được truyền cho CAS server thông qua giao thức HTTPS là một giao thức bảo đảm dữ liệu được mã hóa trước truyền đi.

Xác thực thành công, TGC được sinh ra và thêm vào trình duyệt dưới hình thức là cookie. TGC này sẽ được sử dụng để Single Sign-On với tất cả các ứng dụng.

✚ **Truy cập vào ứng dụng:**

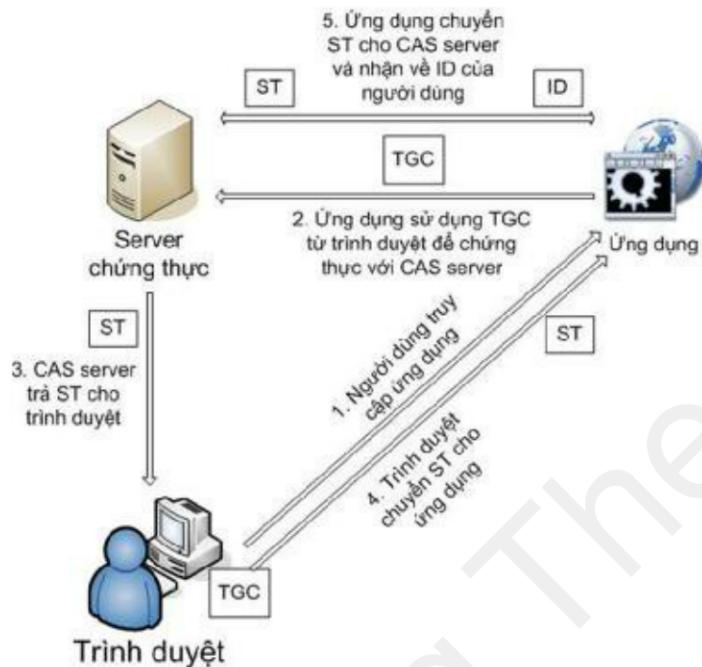
- Người dùng truy cập vào ứng dụng mà chưa chứng thực với CAS server :



Hình 1.2 Người dùng truy cập vào ứng dụng mà chưa chứng thực với CAS server

- Người dùng truy xuất ứng dụng thông qua trình duyệt. Vì chưa nhận được TGC nên ứng dụng sẽ chuyển hướng người dùng cho CAS server.
- Người dùng cung cấp Username / Password của mình thông qua khung đăng nhập để CAS xác thực. Thông tin được truyền đi thông qua giao thức HTTPS.
- Xác thực thành công, CAS server sẽ trả về cho trình duyệt đồng thời TGC và Service Ticket (ST).
- Trình duyệt sẽ giữ lại TGC để sử dụng cho các ứng dụng khác (nếu có) và truyền ST cho ứng dụng.
- Ứng dụng chuyển ST cho CAS server và nhận về ID của người dùng.
- Ứng dụng đăng nhập cho người dùng và bắt đầu phục vụ người dùng.

➤ Người dùng truy cập vào ứng dụng khi đã chứng thực với CAS server :



Hình 1.3 Người dùng truy cập vào ứng dụng khi đã chứng thực với CAS server

- Người dùng truy xuất ứng dụng thông qua trình duyệt.
- Ứng dụng lấy TGC từ trình duyệt và chuyển nó cho CAS server thông qua giao thức HTTPS.
- Nếu TGC này là hợp lệ, CAS server trả về một Service Ticket cho trình duyệt, trình duyệt truyền ST vừa nhận cho ứng dụng.
- Ứng dụng sử dụng ST nhận được từ trình duyệt và sau đó chuyển nó cho CAS server.
- CAS server sẽ trả về ID của người dùng cho ứng dụng, mục đích là để thông báo với ứng dụng người dùng này đã được chứng thực bởi CAS server.
- Ứng dụng đăng nhập cho người dùng và bắt đầu phục vụ người dùng.

2.1.4.4 CAS Uniform Resource Identifiers (CAS URIs):

CAS là một giao thức dựa trên HTTP, yêu cầu mỗi thành phần của được truy xuất thông URI cụ thể. Trong phần này sẽ mô tả về từng URI.

A. /login:

Giao tác URI /login với 2 hành vi: như một người yêu cầu thông tin chứng thực (credential requestor), và như người chấp thuận thông tin chứng thực (credential acceptor).

 /login như một credential requestor:

Nếu client đã thiết lập phiên SSO với CAS, trình duyệt web chuyển đến CAS một cookie bảo mật chứa chuỗi định danh và ticket-granting ticket. Cookie này được gọi là ticket-granting cookie. Nếu các khóa ticket-granting cookie phù hợp với ticket-granting ticket, CAS cung cấp một service ticket. Ticket-granting cookie sẽ được tìm hiểu trong phần Thực Thể CAS.

➤ *Tham số:*

Các tham số HTTP yêu cầu sau có thể được thông qua /login, trong khi hành động như một credential requestor. Những tham số này phân biệt hoa thường, nên phải được xử lý qua /login.

- **service** [tùy chọn]: định danh của ứng dụng mà client truy xuất đến. Trong hầu hết các trường hợp, thì tham số này sẽ là Uniform Resource Locator (URL) của ứng dụng.
- **renew** [tùy chọn]: tham số này thiết lập, phiên SSO đang tồn tại sẽ được bỏ qua. Trong trường hợp này, CAS yêu cầu client trình thông tin chứng thực bắt kể sự tồn tại của phiên SSO với CAS. Tham số này với tham số GATEWAY không tương thích với nhau. Hành vi không được xác định rõ nếu cả 2 được thiết lập. Lời đề nghị, là khi thiết lập tham số “renew” là TRUE thì CAS thực hiện loại bỏ tham số “gateway”.
- **gateway** [tùy chọn]: tham số này được thiết lập thì CAS sẽ không hỏi về bất kỳ thông tin gì từ phía Client. Nếu client đã tồn tại phiên SSO với CAS từ trước hay phiên SSO được thiết lập trên cơ sở xác thực tin cậy (trust authentication), CAS sẽ chuyển client đến URL được chỉ định qua tham số “service”, được gắn thêm vào một giá trị service ticket. (trường hợp này có thể CAS sẽ chèn vào một trang thông báo cho client rằng xác thực CAS đã và đang diễn ra). Ngược lại, nếu client không tồn tại phiên SSO nào với CAS, và cũng không thể thiết lập trust authentication, thì CAS sẽ chuyển client đến URL được chỉ định qua tham số “service” và không có tham số “ticket” được kèm theo URL. Nếu tham số “service” không được chỉ định

và “gateway” được thiết lập, thì hành vi của CAS không được xác định. Giá trị “gateway” khi thiết lập là TRUE.

- URL ví dụ về /login:

Url đăng nhập bình thường:

<https://server/cas/login?service=http%3A%2F%2Fwww.service.com>

Url không nhắc nhớ username/password:

<https://server/cas/login?service=http%3A%2F%2Fwww.service.com&gateway=true>.

Url luôn nhắc nhớ username/password:

<https://server/cas/login?service=http%3A%2F%2Fwww.service.com&renew=true>.

- *Hồi đáp (response) sự chứng thực username/password:*

Khi hành vi /login như một credential requestor, hồi đáp khác nhau với từng loại thông tin chứng thực (credential) được yêu cầu. Nhưng thường thì CAS sẽ trả về một giao diện đăng nhập yêu cầu username và password. Giao diện này sẽ thêm vào một form với các tham số như “username”, “password”, “lt”, hay có thể có thêm tham số “warn”. Nếu “service” được chỉ định rõ đến /login, thì “service” cũng sẽ là một tham số trong form, chứa giá trị chuyển đến /login. (sẽ hiểu rõ hơn các tham số trên ở phần).

Form sẽ được gửi thông qua phương thức POST của HTTP đến /login.

- *Response sự chứng thực tin cậy (trust authentication):*

Trust authentication cung cấp sự xem xét trên khía cạnh nào đó của yêu cầu như là cơ sở cho sự chứng thực.

Chứng thực tin cậy là nó sẽ xác định loại thông tin chứng thực nào sẽ nhận. Nếu thông tin chứng thực thích hợp, thì CAS sẽ chuyển người dùng đến dịch vụ tương ứng mà người dùng không hay biết hay hiển thị thông báo rằng đã nhận được thông tin chứng thực như vậy và cho phép người dùng xác nhận rằng CAS muốn sử dụng thông tin chứng thực đó. Ngược lại, khi không có thông tin chứng thực thích hợp hay không tồn tại một thông tin chứng thực nào, thì CAS sẽ thông báo lý do chứng thực thất bại, và đề nghị với người dùng một sự chứng thực thay thế (chẳng hạn như chứng thực username/password).

- *Response sự chứng thực một cửa (Single Sign-On)*

Nếu client đã và đang thiết lập một phiên làm việc SSO với CAS, thì client sẽ chuyển cookie phiên làm việc HTTP đến /login.

➤ /login như một credential acceptor:

Khi một tập thông tin chứng thực được chấp thuận thông qua /login, /login hành động như credential acceptor.

➤ *Tham số chung:*

Các tham số HTTP yêu cầu sau có thể được thông qua /login, trong khi hành động như một credential acceptor. Những tham số này phân biệt hoa thường, nên phải được xử lý qua /login.

- service [tùy chọn]: URL của ứng dụng mà client truy xuất. CAS sẽ chuyển client đến URL sau khi chứng thực thành công.
- warn [tùy chọn]: tham số này được thiết lập thì Single Sign-On không trở nên trong suốt. Mà client phải được nhắc nhở trước khi bắt đầu chứng thực đến một dịch vụ khác.

➤ *Tham số chứng thực username/password:*

Ngoài các tham số chung, các tham số sau sẽ bắt buộc phải có trong chứng thực username/password, các tham số này phân biệt hoa thường:

- username [bắt buộc]: tên đăng nhập của client dùng để đăng nhập.
- password [bắt buộc]: mật khẩu của client dùng để đăng nhập.
- lt [bắt buộc]: là một login ticket. Chi tiết của tham số này sẽ được hiểu rõ trong phần

➤ *Response:*

- Đăng nhập thành công: chuyển client đến URL được chỉ định bởi tham số “service” theo cách mà các thông tin chứng thực của client không chuyển đến dịch vụ. Sự chuyển tiếp là kết quả của phương thức GET mà client phát ra để yêu cầu dịch vụ. Yêu cầu bao gồm một giá trị service ticket, thông qua tham số “ticket”. Nếu “service” không được chỉ định rõ, CAS sẽ hiển thị thông báo phiên làm việc SSO khởi tạo thành công.
- Đăng nhập thất bại: trả về /login như một credential requestor. CAS sẽ hiển thị một thông báo mô tả tại không đăng nhập thất bại (ví dụ: sai mật khẩu, tài khoản bị khóa,...), và cung cấp một cơ hội khác cho người dùng cố gắng đăng nhập lại.

B. /logout:

/logout hủy phiên làm việc SSO với CAS. Ticket Grating cookie bị hủy, tiếp chuyền đến /login và sẽ không có được service ticket cho đến khi người dùng cung cấp lại thông tin chứng thực (và qua đó thiết lập lại phiên làm việc SSO).

➤ *Tham số:*

- url [tùy chọn]: tham số này chỉ định đến URL của trang mà nó sẽ hiển thị sau khi đăng xuất thành công.

➤ *Response:*

/logout hiển thị trạng thái mà người dùng đã đăng xuất.

C. /validate [CAS 1.0]:

/validate xác nhận tính hợp lệ của dịch vụ. CAS xác nhận service ticket với service ticket được tạo ra từ thông tin chứng thực mà được thu thập từ request, và nếu thất bại thì sẽ hiển thị một response báo thất bại. Ngược lại, nếu thành công thì /validate sẽ trả về username cùng với response. /validate là một phần của giao thức CAS 1.0 do đó nên không xử lý chứng thực proxy. CAS sẽ trả về một response thất bại về sự xác nhận ticket khi một proxy ticket được gửi đến /validate.

➤ *Tham số:*

- service [bắt buộc]: định danh của dịch vụ (thường là URL của ứng dụng).
- ticket [bắt buộc]: service ticket được cấp phát bởi /login.
- renew [tùy chọn]: nếu tham số này được thiết lập, sự xác nhận ticket chỉ thành công khi service ticket được cấp phát từ việc việc trình bày thông tin chứng thực của người dùng (từ giao diện đăng nhập hoặc chứng thực tin cậy, nơi tạo nên phiên làm việc SSO). Sự xác nhận ticket thất bại khi ticket được cung cấp từ phiên làm việc SSO.

➤ *Response:*

/validate sẽ trả về một trong hai dạng response sau:

- Xác nhận ticket thành công:

yes<LF>

username<LF>

- Xác nhận ticket thất bại:

no<LF>
<LF>

➤ Ví dụ URL của /validate:

- Xác nhận đơn giản:

<https://server/cas/validate?service=http%3A%2F%2Fwww.service.com&ticket=ST-1856339-aA5Yuvrxzpv8Tau1cYQ7>

- Đảm bảo service ticket được cung cấp bởi sự trình bày thông tin chứng thực của người dùng:

<https://server/cas/validate?service=http%3A%2F%2Fwww.service.com&ticket=ST-1856339-aA5Yuvrxzpv8Tau1cYQ7&renew=true>

D. /serviceValidate [CAS 2.0]:

/serviceValidate xác nhận tính hợp lệ của service ticket và trả về response. /serviceValidate làm việc tương tự như /validate và cũng tạo được proxy-granting ticket nếu cần thiết. Nếu /serviceValidate nhận được một proxy ticket thì sẽ không trả về chứng thực thành công, /serviceValidate sẽ gửi kèm một thông điệp báo lỗi vào response. CAS cũng xác minh định danh dịch vụ (service identifier) với những dịch vụ đã được đăng ký, và nếu không có dịch vụ đăng ký nào phù hợp với định danh dịch vụ thì một response báo lỗi sẽ được gửi trả lại. /serviceValidate là phần mở rộng của /validate trong CAS 2.0.

➤ Tham số:

Ngoài 3 tham số tương tự như /validate. /serviceValidate có thêm một vài tham số tùy chọn sau:

- pgturl [tùy chọn]: url của proxy callback. Nếu một dịch vụ muốn ủy nhiệm một chứng thực client đến dịch vụ đầu cuối (back-end service), thì phải có được proxy-granting ticket. Để có được ticket thì phải được xử lý thông qua một URL proxy callback. URL này là duy nhất và an toàn để nhận dạng dịch vụ đầu cuối rằng đang ủy nhiệm cho sự chứng thực của client. Dịch vụ đầu cuối có thể quyết định dù có hay không chấp nhận

thông tin chứng thực dựa trên URL callback định danh của dịch vụ đầu cuối. Cơ chế Proxy callback được giải thích trong phần Proxy-granting Ticket.

➤ *Response:*

/serviceValidate trả về một CAS serviceResponse với định dạng XML. Sau đây là 2 ví dụ về response mà /serviceValidate trả về.

- Kiểm tra ticket thành công:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationSuccess>
        <cas:user>username</cas:user>
        <cas:proxyGrantingTicket>PGTIOU-84678-
8a9d...</cas:proxyGrantingTicket>
    </cas:authenticationSuccess>
</cas:serviceResponse>
```

- Kiểm tra ticket thất bại:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationFailure code="INVALID_TICKET">
        Ticket ST-1856339-aA5Yuvrxzpv8Tau1cYQ7 not recognized
    </cas:authenticationFailure>
</cas:serviceResponse>
```

➤ *Mã lỗi:*

Các giá trị sau có thể xem như thuộc tính “code” trong response báo chứng thực thất bại. Sau đây là một vài mã lỗi điển hình:

- INVALID_REQUEST: chưa đủ các tham số bắt buộc.
- INVALID_TICKET: ticket được cung cấp không hợp lệ, hay ticket không được khởi tạo từ trang login và “renew” được thiết lập để kiểm tra. Phần thân của khối <cas:authenticationFailure> trong XML response sẽ miêu tả chi tiết một cách chính xác.
- INVALID_SERVICE: ticket cung cấp không hợp lệ, service được chỉ định không phù hợp với service liên quan đến ticket. CAS làm mất hiệu lực ticket đó và sau này sẽ không cho phép kiểm tra ticket giống như thế.
- INTERNAL_ERROR: một lỗi xảy ra trong quá trình xác nhận tính hợp lệ của ticket.

Tất cả những mã lỗi trên, CAS cung cấp thông điệp chi tiết hơn như phần body khôi <cas:authenticationFailure> của XML response.

➤ *Proxy callback:*

Sau đây là cơ chế làm việc của proxy callback:

1. Service yêu cầu một proxy-granting ticket được xác định sau khi xác nhận tính hợp lệ của service ticket hay proxy ticket thông qua /serviceValidate (hay /proxyValidate) kiểm tra tham số “pgturl”. Đây là URL callback của dịch vụ mà CAS sẽ kết nối để xác minh định danh của dịch vụ. URL này phải là HTTPS, và CAS xác minh tính hợp lệ cả chứng nhận SSL (SSL certificate) và URL phải phù hợp với dịch vụ. Nếu xác nhận tính hợp lệ của chứng nhận bị thất bại, thì proxy-granting ticket sẽ không được cấp phát, và response service CAS được mô tả trong phần Response của /serviceValidate sẽ chứa thêm khôi <proxyGrantingTicket>. Tại thời điểm này, việc cấp phát proxy-granting ticket bị dừng lại, nhưng sự xác nhận tính hợp lệ của service ticket vẫn tiếp tục, sẽ trả về thành công hoặc thất bại. Ngược lại, nếu xác nhận tính hợp lệ của chứng nhận là thành công, thì việc cấp phát proxy-granting ticket được xử lý và chuyển tiếp bước 2.
2. CAS sử dụng phương thức GET của HTTP để bỏ qua 2 tham số “pgt” và “pgtIou” (hai tham số này sẽ được nói rõ trong phần CAS Ticket) để lấy “pgturl”.
3. Nếu phương thức GET của HTTP trả về một mã trạng thái HTTP là 200 (Tốt), thì CAS trả lời cho yêu cầu /serviceValidate (hoặc /proxyValidate) với một service response (được mô tả trong phần response của /serviceValidate) chứa đựng proxy-granting ticket IOU bên trong khôi <cas:proxyGrantingTicket>. Nếu phương thức GET của HTTP trả về bất kỳ một mã trạng thái nào khác, ngoài trừ mã chuyển 3xx HTTP, CAS phải trả lời cho yêu cầu /serviceValidate (hoặc /proxyValidate) với một service response không chứa khôi <cas:proxyGrantingTicket>. CAS có thể làm theo bất kỳ chuyển hướng HTTP nào được cung cấp pgtUrl. Tuy nhiên, URL callback đang xác định được cung cấp sau khi xác nhận tính hợp lệ của khôi <proxy> giống như URL ban đầu chuyển đến /serviceValidate (hay /proxyValidate) như là tham số “pgtUrl”.

4. Các dịch vụ, nhận được proxy-granting ticket IOU trong CAS response, và cả hai proxy-granting ticket và proxy-granting ticket IOU đều từ proxy callback, sẽ sử dụng proxy-granting ticket IOU để có thể tương quan proxy-granting ticket với response xác nhận tính hợp lệ. Sau đó các dịch vụ sử dụng proxy-granting ticket cho việc đạt được proxy ticket.

➤ *Ví dụ URL cho /serviceValidate:*

- Xác nhận tính hợp lệ đơn giản:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com
&ticket= ST-1856339-aA5Yuvrxzpv8Tau1cYQ7
```

- Đảm bảo service ticket được cấp phát bằng việc trình thông tin chứng thực:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com
&ticket= ST-1856339-aA5Yuvrxzpv8Tau1cYQ7&renew=true
```

- Thông qua URL callback cho vấn đề ủy nhiệm:

```
https://server/cas/serviceValidate?service=http%3A%2F%2Fwww.service.com
&ticket= ST-1856339-aA5Yuvrxzpv8Tau1cYQ7
&pgtUrl=https://my-server/myProxyCallback
```

E. /proxyValidate: [CAS 2.0]

/proxyValidate thực hiện nhiệm vụ xác nhận tính hợp lệ tương tự như /serviceValidate và thêm vào đó là xác nhận tính hợp lệ proxy ticket. /proxyValidate có khả năng xác nhận tính hợp lệ của cả service ticket và proxy ticket.

➤ *Tham số:*

/proxyValidate có các tham số giống như các tham số của /serviceValidate.

➤ *Response:*

/proxyValidate cũng trả về một CAS serviceResponse với định dạng XML. Sau đây là 2 ví dụ về response mà /proxyValidate trả về.

- Xác nhận tính hợp lệ của ticket thành công:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
  <cas:authenticationSuccess>
    <cas:user>username</cas:user>
```

```
<cas:proxyGrantingTicket>
    PGTIOU-84678-8a9d...
</cas:proxyGrantingTicket>
<cas:proxies>
    <cas:proxy>https://proxy2/pgtUrl</cas:proxy>
    <cas:proxy>https://proxy1/pgtUrl</cas:proxy>
</cas:proxies>
</cas:authenticationSuccess>
</cas:serviceResponse>
```

Lưu ý: khi chứng thực qua nhiều proxy, thì thứ tự các proxy đã đi qua được phản ánh qua khái `<cas:proxies>`. Proxy nào đi qua gần nhất sẽ nằm ở đầu danh sách, khi có proxy mới thêm vào thì danh sách sẽ chuyển xuống để thêm vào đầu danh sách. Ở ví dụ trên, dịch vụ được xác định bởi <https://proxy1/pgtUrl> đã được đi qua đầu tiên, và dịch vụ được ủy nhiệm chứng thực dịch vụ được xác định bởi <https://proxy2/pgtUrl>.

- Xác nhận tính hợp lệ của ticket thất bại:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:authenticationFailure code="INVALID_TICKET">
        ticket PT-1856376-1HMgO86Z2ZKeByc5XdYD not recognized
    </cas:authenticationFailure>
</cas:serviceResponse>
```

➤ *Ví dụ URL cho /proxyValidate:*

/proxyValidate sử dụng các tham số giống với /serviceValidate, nên có thể thay đổi các tham số của /serviceValidate và thay thế “serviceValidate” thành “proxyValidate”.

F. /proxy [CAS 2.0]:

/proxy cung cấp proxy ticket đến dịch vụ yêu cầu proxy-granting ticket và ủy nhiệm chứng thực các dịch vụ đầu cuối.

➤ *Tham số:*

- pgt [bắt buộc]: là proxy-granting ticket được yêu cầu bởi dịch vụ trong suốt quá trình xác nhận tính hợp lệ của service ticket hay proxy ticket.
- targetService [bắt buộc]: định dạng dịch vụ của dịch vụ đầu cuối. Lưu ý rằng không phải tất cả các dịch vụ đầu cuối đều là dịch vụ web, do đó định danh dịch vụ không phải luôn luôn là một URL. Tuy nhiên, định danh dịch

vụ được chỉ định ở đây phải phù hợp với tham số “service” được chuyển cho /proxyValidate sau khi xác nhận tính hợp lệ của proxy ticket.

➤ *Response:*

- Yêu cầu thành công:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:proxySuccess>
        <cas:proxyTicket>
            PT-1856392-b98xZrQN4p90ASrw96c8
        </cas:proxyTicket>
    </cas:proxySuccess>
</cas:serviceResponse>
```

- Yêu cầu thất bại:

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
    <cas:proxyFailure code="INVALID_REQUEST">
        'pgt' and 'targetService' parameters are both required
    </cas:proxyFailure>
</cas:serviceResponse>
```

➤ *Mã lỗi:*

- INVALID_REQUEST: chưa đầy đủ các tham số bắt buộc.
- BAD_PGT: pgt được cung cấp không hợp lệ.
- INTERNAL_ERROR: một lỗi xảy ra trong quá trình xác nhận tính hợp lệ của ticket.

Tất cả những mã lỗi trên, CAS cung cấp thông điệp chi tiết hơn như phần body khối <cas:authenticationFailure> của XML response.

➤ *Ví dụ URL về /proxy:*

Yêu cầu proxy đơn giản:

[https://server/cas/proxy?targetService=http%3A%2F%2Fwww.service.com
&pgt=PGT-490649-W81Y9Sa2vTM7hda7xNTkezTbVge4CUSybAr...](https://server/cas/proxy?targetService=http%3A%2F%2Fwww.service.com&pgt=PGT-490649-W81Y9Sa2vTM7hda7xNTkezTbVge4CUSybAr...)

2.1.4.5 Thực thể CAS (CAS Entities):

✚ **Service Ticket:**

Service Ticket là một chuỗi mờ (opaque string) được client sử dụng như một thông tin chứng thực để có quyền truy xuất đến dịch vụ. Service Ticket thu

được từ CAS sau khi client trình thông tin chứng thực và định danh dịch vụ đến /login.

Tính chất của Service Ticket:

- Service Ticket chỉ hợp lệ với định danh dịch vụ, khi được tạo ra thì định danh dịch vụ được chỉ định đến /login. Định danh dịch vụ không là thành phần của Service Ticket.
- Service Ticket chỉ hợp lệ với ticket đã được xác nhận tính hợp lệ. Cho dù kiểm tra có thành công hay không, sau khi thực hiện kiểm tra thì ticket đó sẽ mất hiệu lực, vì sau này xác nhận tính hợp lệ các ticket giống như vậy xem như thất bại.
- CAS làm mất hiệu lực những Service Ticket không được xác nhận tính hợp lệ vì một lý do hết thời gian sau khi được cấp phát. Nếu một dịch vụ chuyển cho quá trình xác nhận tính hợp lệ một Service Ticket đã hết hạn, thì CAS sẽ trả về một response báo xác nhận thất bại.
- Service Ticket chứa dữ liệu ngẫu nhiên đủ bảo mật để một ticket không có thể đoán được.
- Service Ticket bắt đầu với các ký tự “ST-”.
- Dịch vụ có thể chấp nhận Service Ticket có chiều dài lên đến 32 ký tự.

Proxy Ticket:

Mô tả tóm tắt về proxy: Proxy là một cái gì đó hoạt động như là 1 server, nhưng khi được yêu cầu từ client, bản thân nó hoạt động như là 1 client truy xuất đến máy chủ thật sự. Một HTTP proxy không chuyển tiếp mọi yêu cầu qua nó. Thay vào đó, đầu tiên nó kiểm tra trang web được yêu cầu có ở trong bộ nhớ cache của nó. Nếu có, nó sẽ trả về trang web đó mà không cần gửi yêu cầu đến máy chủ đích. Bởi vì proxy hoàn toàn định giới hạn các kênh giao tiếp, proxy được xem là một kỹ thuật tường lửa an ninh hơn là bộ lọc gói, vì chúng làm tăng đáng kể sự cô lập giữa các mạng.

Trong CAS, proxy là một dịch vụ, truy xuất đến một dịch vụ khác với tư cách là một người dùng cụ thể. Proxy Ticket được tạo ra từ CAS sau khi một dịch vụ đưa ra một proxy-granting ticket hợp lệ, và một định danh dịch vụ (giá trị của tham số “service” của url /proxy) cho dịch vụ đầu cuối đang được kết nối.

Tính chất của Proxy Ticket:

- Các tính chất tương tự như Service Ticket, chỉ việc thay đổi “Service Ticket” thành “Proxy Ticket”.
- Proxy Ticket bắt đầu với các ký tự “PT-”.

Proxy-Granting Ticket:

Proxy-Granting Ticket là một chuỗi mờ được dịch vụ sử dụng để có được Proxy Ticket để có quyền truy xuất đến các dịch vụ đầu cuối với tư cách là một client. Proxy-Granting Ticket có được từ CAS sau khi xác nhận tính hợp lệ của Service Ticket hay Proxy Ticket. Sự cung cấp Proxy-Granting Ticket đã được mô tả trong phần Proxy Callback ở trên.

Tính chất của Proxy-Granting Ticket:

- Proxy-Granting Ticket được các dịch vụ sử dụng để có được nhiều Proxy Ticket. Proxy-Granting Ticket là ticket không sử dụng 1 lần.
- Proxy-Granting Ticket sẽ hết hạn khi sự chứng thực của client đang bắt đầu ủy nhiệm đăng xuất khỏi CAS.
- Proxy-Granting Ticket chứa dữ liệu ngẫu nhiên đủ an toàn để ticket không thể được đoán ra với một khoảng thời gian hợp lý khi bị tấn công vũ lực (brute-force attack).
- Proxy-Granting Ticket bắt đầu bằng các ký tự “PGT-”.
- Dịch vụ có thể xử lý Proxy-Granting Ticket có chiều dài lên đến 64 ký tự.

Proxy-Granting Ticket IOU:

Proxy-Granting Ticket IOU (PGTIOU) là một chuỗi mờ được đặt trong response được cung cấp bởi /serviceValidate và /proxyValidate sử dụng để liên kết một Service Ticket hay một Proxy Ticket với một Proxy-Granting Ticket cụ thể. Xử lý này cũng đã được mô tả qua phần Proxy Callback.

Tính chất Proxy-Granting Ticket IOU:

- Proxy-Granting Ticket IOU không chứa bất kỳ một tham chiếu nào đến Proxy-Granting Ticket được kết hợp. Truyền một PGTIOU cụ thể, PGTIOU không thể bị tìm ra Proxy-Granting Ticket tương ứng bằng các phương thức thuật toán trong một khoảng thời gian hợp lý.
- PGTIOU chứa dữ liệu ngẫu nhiên đủ an toàn để một ticket không thể được đoán ra với một khoảng thời gian hợp lý khi bị tấn công vũ lực (brute-force attack).
- Proxy-Granting Ticket IOU bắt đầu với các ký tự “PGTIOU-”.

- Dịch vụ có thể xử lý PGTIOU có chiều dài lên đến 64 ký tự.

Login Ticket:

Login Ticket là một chuỗi được cung cấp bởi /login với hoạt động như là một credential requestor và thông qua /login với hoạt động như là một credential acceptor cho chứng thực username/password. Mục đích của Login Ticket là ngăn cấm việc phát lại thông tin chứng thực do lỗi trình duyệt.

Tính chất của Login Ticket:

- Login Ticket được cấp phát bởi /login phải có được xác xuất duy nhất.
- Login Ticket chỉ hợp lệ cho một sự nỗ lực chứng thực. Dù chứng thực có thành công hay không, thì CAS cũng làm mất hiệu lực Login Ticket, vì sau này những sự nỗ lực chứng thực với thẻ hiện của Login Ticket đều thất bại.
- Login Ticket bắt đầu với các ký tự “LT-”.

Ticket-Granting Cookie:

Ticket-Granting Cookie là một HTTP Cookie được thiết lập bởi CAS sau khi thiết lập một phiên làm việc SSO. Cookie này duy trì tình trạng đăng nhập cho client, trong khi nó còn hợp lệ, client có thể đưa nó cho CAS để thay cho thông tin chứng thực. Dịch vụ có thể bỏ qua SSO bằng cách thiết lập tham số “renew” như đã nêu ở các phần tham số của /login, /validate, /serviceValidate.

Tính chất của Ticket-Granting Cookie:

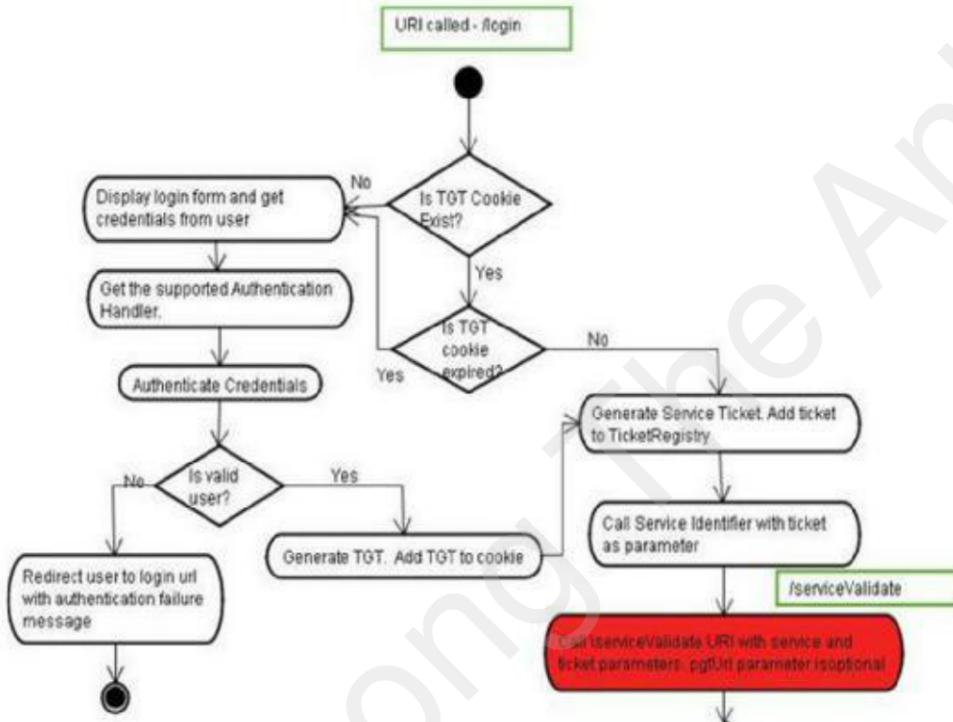
- Ticket-Granting Cookie phải được thiết lập hết hạn khi kết thúc phiên làm việc của trình duyệt.
- CAS thiết lập đường dẫn cookie là giới hạn. Ví dụ: Nếu CAS Server thiết lập theo đường dẫn /cas, thì đường dẫn cookie phải được thiết lập đến /cas.
- Giá trị của Ticket-Granting Cookie chứa dữ liệu ngẫu nhiên đủ an toàn để một ticket-granting cookie không thể được đoán ra trong khoảng thời gian hợp lý.
- Giá trị của Ticket-Granting Cookie bắt đầu với các ký tự “TGC-”.

Tập ký tự Ticket và Ticket-Granting Cookie:

Ngoài các yêu cầu trên, tất cả CAS Ticket và giá trị của Ticket-Granting Cookie chỉ chứa các ký tự trong tập sau { A-Z, a-z, 0-9, và ký tự gạch nối “-” }.

2.1.4.6 Kiến trúc CAS:

✚ Luồng đăng nhập (Login Flow):

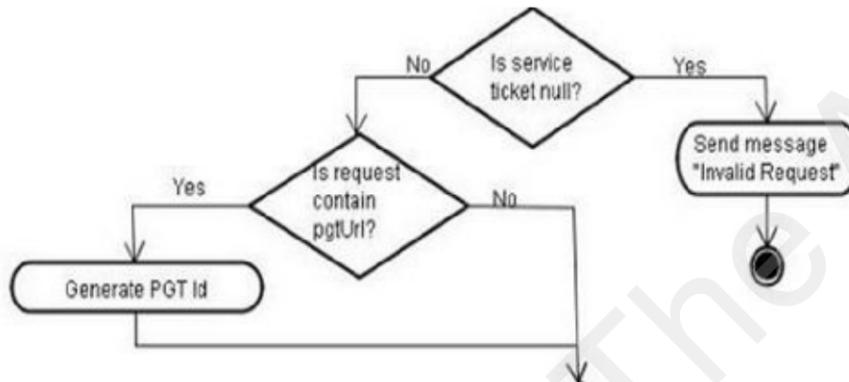


Hình 1.4 CAS - luồng đăng nhập (1)

CAS có thể truy xuất thông qua nhiều URI khác nhau. URI chủ yếu là /login. Khi người dùng nhập địa chỉ <https://localhost:8443/cas-web/login> từ trình duyệt, CAS sẽ kiểm tra Ticket-Granting Cookie (TGC) đã tồn tại chưa. Nếu TGC đã tồn tại, thì nó sẽ kiểm tra thời gian hết hạn của cookie và nếu còn thời hạn thì Service Ticket (ST) sẽ được sinh ra. Nếu TGC không tồn tại hay đã hết hạn thì CAS sẽ buộc người dùng nhập thông tin đăng nhập vào giao diện đăng nhập.

Người dùng nhập thông tin đăng nhập và chọn Submit. CAS sẽ lấy danh sách AuthenticationHandlers (trình xử lý chứng thực) từ tập tin deployerConfigContext.xml và kiểm tra xem nó hỗ trợ trình xử lý chứng thực nào. CAS sẽ đưa thông tin đăng nhập cho trình xử lý chứng thực mà nó hỗ trợ và kiểm tra thông tin đăng nhập của người dùng. Nếu người dùng chứng thực không hợp lệ sẽ được chuyển đến khung đăng nhập để đăng nhập lại. Nếu người dùng hợp lệ thì một Ticket-Granting Ticket (TGT) sẽ được sinh ra và thêm vào cookie.

CAS tạo ra một ST và thêm ST này đến chỗ đăng ký Ticket (Ticket Registry). CAS kiểm tra nếu tham số dịch vụ được đưa với URL /login trong đó tên tham số là “service” và giá trị của tham số là URL của dịch vụ. Ví dụ: <https://localhost:8443/cas-web/login?service=http://localhost:8080/c/portal/login>.



Hình 1.5 CAS - luồng đăng nhập (2)

CAS gọi Service Identifier với ticket là tham số, giá trị là ST. Client có trách nhiệm gọi /serviceValidate URI của CAS để xác nhận service và service ticket. Thường /serviceValidate được gọi bởi lớp Filter trong mã của client. /serviceValidate sẽ được gọi cùng với các tham số bắt buộc như service với giá trị định danh dịch vụ, serviceticket với giá trị là ST, và một tham số tùy chọn là pgtUrl với giá trị là proxy-granting URL. Ví dụ: chẳng hạn với /login url ở ví dụ: /serviceValidate?service=http://localhost:8080/c/portal/login&ticket=ST-1856339-aA5Yuvrxzpv8Tau1cYQ7&pgtUrl=https://server/test.jsp

Service param name: **service**

Service Identifier value: <http://localhost:8080/c/portal/login>

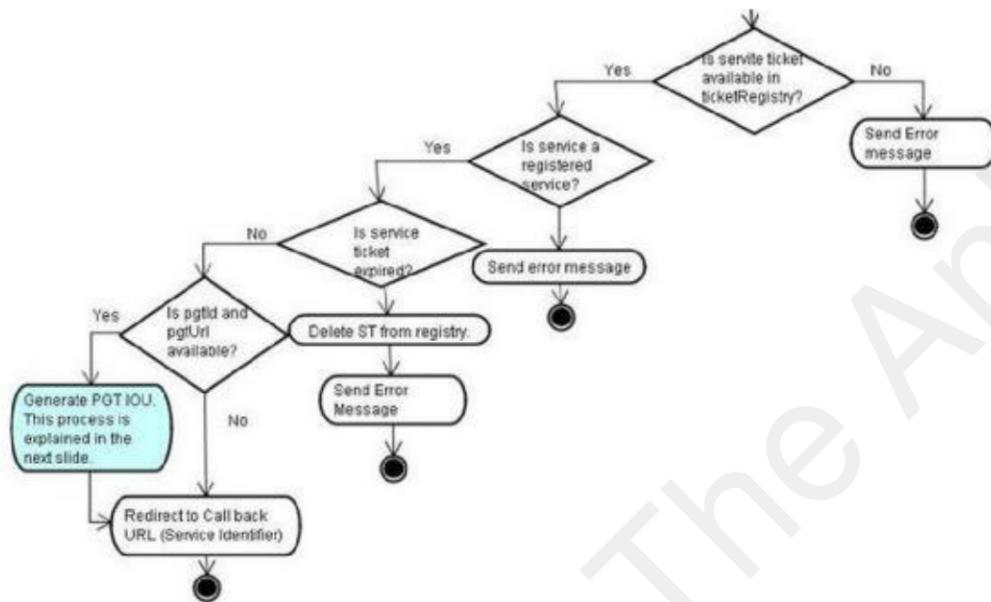
Ticket param name: **ticket**

Service Ticket value: ST-1856339-aA5Yuvrxzpv8Tau1cYQ7

Proxy param name: **pgtUrl**

Proxy-granting Url value: <https://server/test.jsp>

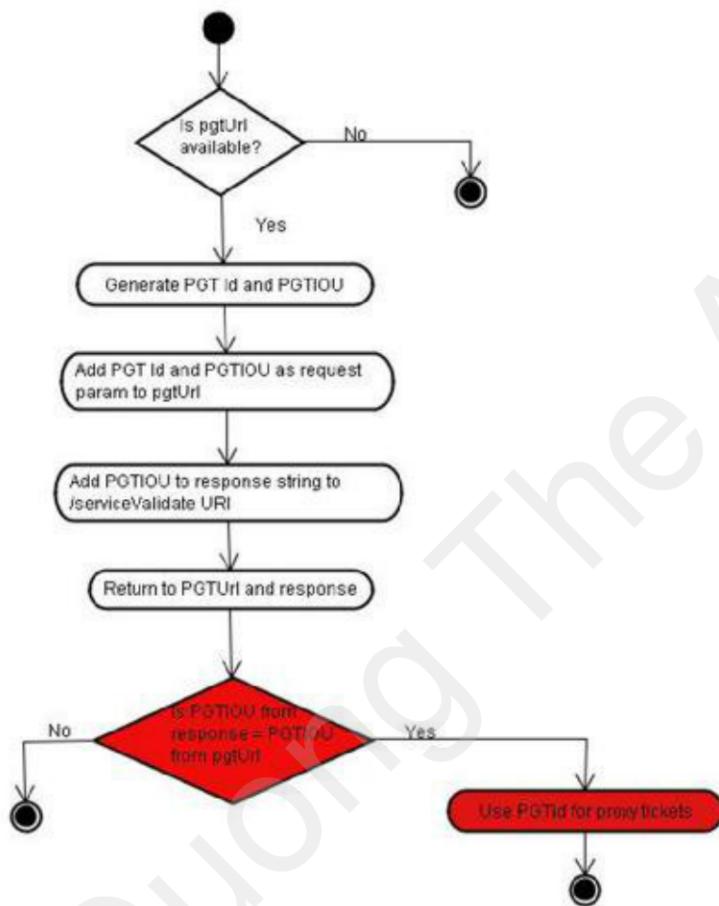
CAS kiểm tra nếu ST là null. Nếu không null, CAS kiểm tra nếu tham số tùy chọn pgtUrl có giá trị hay không. Nếu pgtUrl có giá trị, Proxy-Granting Ticket (PGT) được sinh ra.



Hình 1.6 CAS - luồng đăng nhập (3)

Ké đến, CAS kiểm tra nếu ST đã có được từ tham số bắt buộc đã có sẵn trong nơi đăng ký dịch vụ (service registry) chưa. Nếu đã có, CAS kiểm tra xem dịch vụ là có đăng ký chưa. Nếu dịch vụ đã đăng ký, ST được kiểm tra xem còn thời hạn hay không. Nếu đã hết thời hạn, thì ST sẽ bị xóa trong Service Registry và 1 thông điệp báo lỗi sẽ được gửi đến cho client. Nếu bất kỳ một điều kiện nào ở trên không được thỏa, thì sẽ có một thông điệp báo lỗi gửi đến client, và đến client để xử lý lỗi.

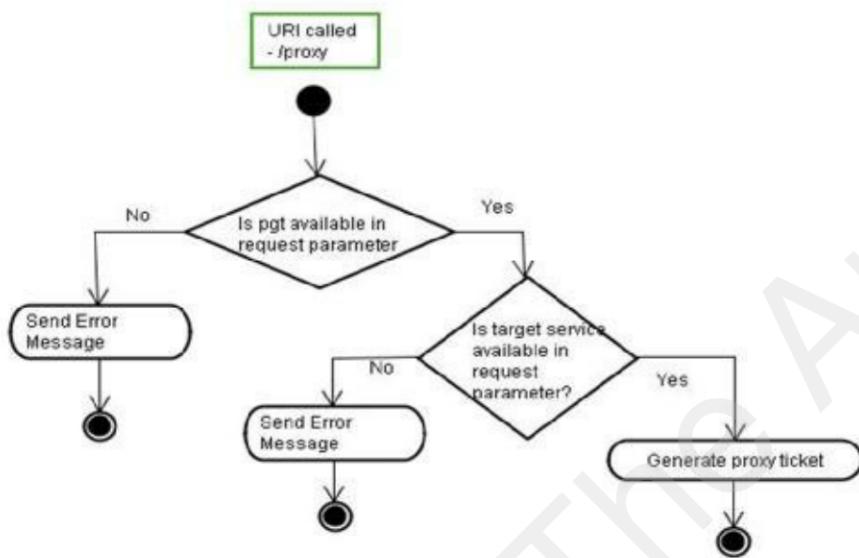
Nếu ST chưa hết thời hạn, thì CAS kiểm tra xem PGT đã tạo ra và tham số pgtUrl có giá trị không. Nếu có, PGTIOU được tạo ra và chuyển đến URL callback.



Hình 1.7 CAS - luồng đăng nhập (4)

Giá trị của tham số dịch vụ với ticket là tên tham số, PGT là giá trị, PGTIOU là tên tham số, PGTIOU vừa được sinh ra là giá trị tham số. CAS add PGTIOU vào chuỗi response và gọi URL callback – pgtUrl. Client phải xác nhận tính hợp lệ, nếu PGTIOU có được từ chuỗi response giống với PGTIOU có được từ tham số yêu cầu PGTIOU (như Hình 1. 7). Nếu tham số pgtUrl không có, thì CAS gọi URL callback – giá trị của tham số yêu cầu “service”. Ví dụ: <https://server/test.jsp?ticket=PGT-123423423-yTresprnts8Tau1cYQ7&PGTIOU=PGTIOU-35356436-hfadYrjagsjhghjgwdfTF9>

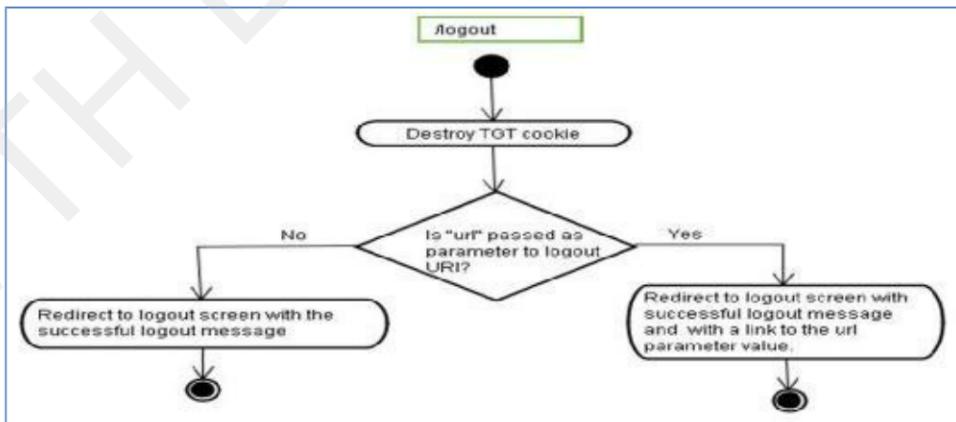
✚ Luồng Proxy (Proxy Flow):



Hình 1.8 CAS - Proxy Flow

Sau khi PGTIOU được client xác nhận tính hợp lệ, client phải gọi URL /proxy với pgt và targetService như là những tham số cho dịch vụ hoạt động như proxy đến bất kỳ dịch vụ đích nào. Tham số pgt chứa pgt Id là giá trị và targetService chứa định danh của dịch vụ đích như là giá trị. CAS kiểm tra xem pgt Id và targetService có sẵn dùng hay không, và nếu có thì Proxy Ticket sẽ được sinh ra.

✚ Luồng đăng xuất (Logout Flow):



Hình 1.9 CAS – Luồng đăng xuất

Khi người dùng muốn đăng xuất khỏi một service, người dùng phải gọi /logout URI. CAS sẽ hủy Ticket-Granting Cookie và kiểm tra /logout URI có chứa tham số url hay không. Nếu tham số “url” có giá trị, một thông báo đăng xuất thành công sẽ được hiển thị với một liên kết đến giá trị url, ngược lại thì chỉ có thông báo đăng xuất thành công.

2.1.4.7 Lợi ích CAS:

- ✓ CAS cung cấp sự nhất quán, an toàn, đáng tin cậy khi xác thực cho các ứng dụng web.
- ✓ CAS tương đối mạnh – là sản phẩm của nhiều trường đại học và tổ chức thương mại, được bảo trì bởi cộng đồng người dùng đang sử dụng.
- ✓ Các nhà phát triển không cần quan tâm nhiều đến việc phát triển một cơ chế xác thực riêng.
- ✓ Một người dùng chỉ cần đăng nhập một lần cho nhiều ứng dụng sử dụng cùng dịch vụ CAS.

2.1.4.8 Hạn chế:

- ❖ CAS chỉ giải quyết xác thực (authenticate), không giải quyết cấp phép (authorization).
- ❖ Không single sign-off.
- ❖ Ít tài liệu tham khảo.

2.2 Lightweight Directory Access Protocol:

2.2.1 Khái niệm:

LDAP (Lightweight Directory Access Protocol) là một chuẩn giúp cho các máy tính và thiết bị mạng có thể truy cập vào các thông tin chung thông qua mạng. LDAP cho client-server có thể truy cập đến các thư mục thông qua mạng máy tính. Ngoài việc cung cấp khả năng tìm kiếm (search) và đọc (read) thông tin, LDAP còn có khả năng thêm (add), cập nhật (update) và xóa (delete) thông tin trong thư mục.

2.2.2 Phương thức hoạt động của LDAP:

Giao thức client/server:

Là một mô hình giao thức giữa một chương trình client chạy trên một máy tính gửi một yêu cầu qua mạng đến cho một máy tính khác đang chạy một chương trình sever (phục vụ), chương trình này nhận lấy yêu cầu và thực hiện. Sau đó nó trả về kết quả cho chương trình client. Ví dụ những giao thức client/server khác là giao thức truyền siêu văn

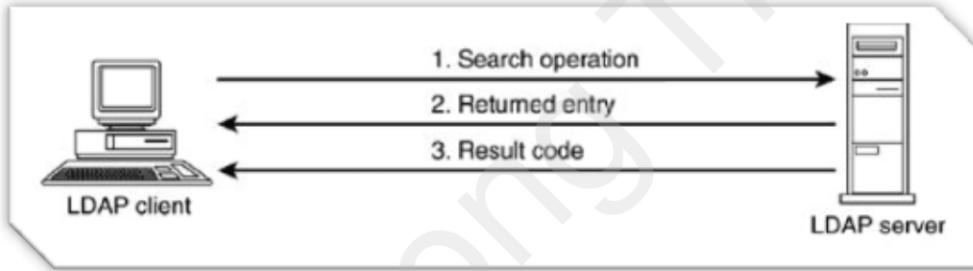
bản (Hypertext transfer protocol) viết tắt là HTTP, giao thức này có những ứng dụng rộng rãi phục vụ những trang web và giao thức Internet Message Access Protocol (IMAP), là một giao thức sử dụng để truy cập đến các thư thông báo điện tử.

Ý tưởng cơ bản của giao thức client/server là công việc được gán cho những máy tính đã được tối ưu hoá để thực hiện. Ví dụ tiêu biểu cho một máy server LDAP có rất nhiều RAM(bộ nhớ) dùng để lưu trữ nội dung các thư mục cho các thao tác thực thi nhanh, và máy này cũng cần đĩa cứng và các bộ vi xử lý ở tốc độ cao.

Giao thức LDAP là giao thức hướng thông điệp:

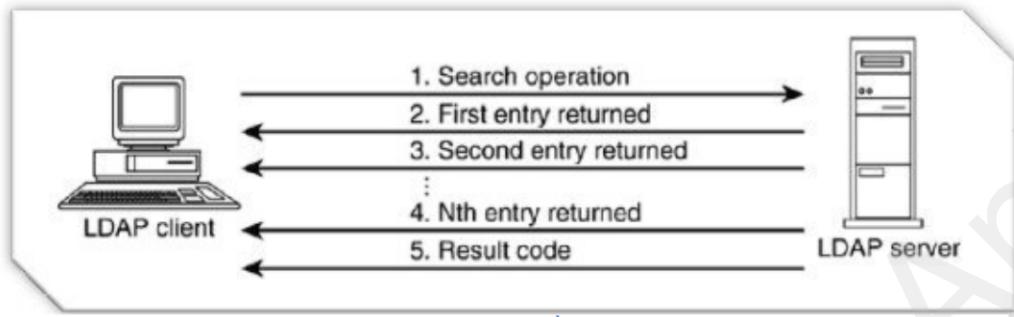
Client xây dựng thông điệp LDAP chứa yêu cầu và gửi nó đến server. Máy chủ xử lý yêu cầu và gửi trả về kết quả cho client như một dãy liên tiếp các thông điệp LDAP.

Ví dụ: khi một client LDAP tìm kiếm thư mục cho một mẫu tin (entry) cụ thể,



Hình 2. 1 Một client nhận một entry từ directory

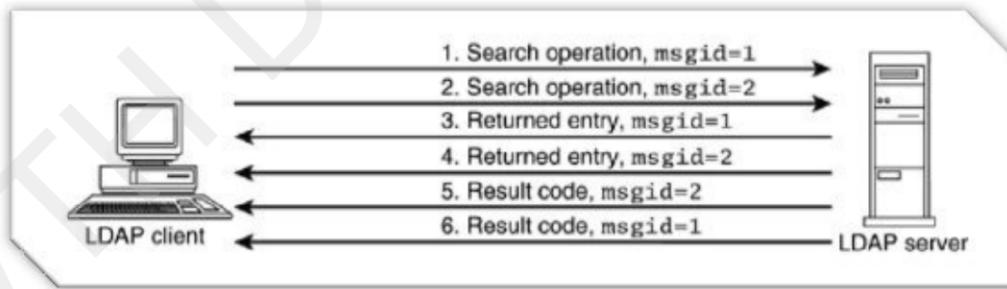
client sẽ gửi một thông điệp LDAP yêu cầu tìm kiếm đến server. Trong thông điệp có chứa một chỉ danh riêng biệt duy nhất, được tạo ra bởi client. Server lấy mẫu tin từ cơ sở dữ liệu và gói trong thông điệp LDAP gửi đến cho client, server cũng trả về mã kết quả đến client trong một thông điệp LDAP riêng. Tất cả những hồi âm từ server đến client đều được xác định với chỉ danh được cung cấp trong thông điệp yêu cầu của client. Hình 2. 1 sẽ thể hiện sự tương tác đó.



Hình 2. 2 Một client nhận nhiều entry từ directory

Nếu client tìm kiếm thư mục và có nhiều mẫu tin được tìm thấy, thì những mẫu tin này sẽ được gửi đến client bằng một dãy liên tiếp các thông điệp LDAP tương ứng một thông điệp cho một mẫu tin. Mỗi một mẫu tin có một tên riêng biệt không trùng lặp, được gọi là DN (Distinguished Name), DN này được gói trong thông điệp LDAP như chuỗi văn bản. Các thông điệp tạo nên kết quả tìm kiếm được kết thúc bằng một thông điệp Result, trong đó bao hàm một kết quả tổng quan cho thao tác tìm kiếm, như Hình 2. 2.

Bởi vì giao thức LDAP là giao thức dựa trên thông điệp, nên client được phép phát nhiều thông điệp yêu cầu cùng một lúc. Ví dụ: một client có thể phát ra đồng thời hai thông điệp yêu cầu tìm kiếm. Client tạo ra cho mỗi một thông điệp một chỉ danh riêng; kết quả trả về cho từng yêu cầu cụ thể sẽ được dán nhãn tương ứng với từng chỉ danh thông điệp, client được phép sắp xếp kết quả trả về đến từng yêu cầu khác nhau sẽ đến theo thứ tự hoặc đồng thời.



Hình 2. 3 Một client phát ra nhiều yêu cầu tìm kiếm đồng thời

Trong Hình 2. 3, client phát ra 2 yêu cầu tìm kiếm đồng thời. server xử lý cả 2 thao tác đó và trả về kết quả cho client. Thông báo rằng server đã gửi mã kết quả của thông điệp 2, được xác định trong hình là $msgid=2$, đến client trước khi server gửi mã kết quả của thông điệp 1. Điều này hoàn toàn chấp nhận được. Những chi tiết

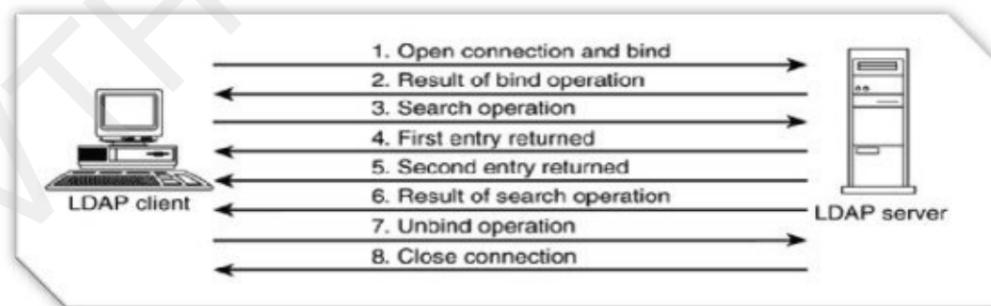
này đều được ẩn đăng sau bởi LDAP SDK (software development kit). Lập trình viên viết ứng dụng LDAP không cần quan đến việc sắp xếp những kết quả này, bởi vì SDK sẽ làm việc này một cách tự động.

Việc cho phép nhiều yêu cầu đồng thời gửi đến server có thể giúp cho LDAP linh hoạt và hiệu quả hơn những giao thức mà kiểu thao tác từng bước (lockstep) chẳng hạn như HTTP (Hypertext Transfer Protocol). Với 1 giao thức lockstep, mỗi một client yêu cầu phải được server trả lời trước khi yêu cầu khác được gửi đến. Ví dụ: một chương trình HTTP client – như trình duyệt web, nếu muốn tải về nhiều tập tin đồng thời thì phải mở những kết nối tương ứng với từng tập tin. LDAP thì khác, có thể quản lý nhiều thao tác trong 1 kết nối, làm giảm được lượng kết nối tối đa đồng thời đến server chuẩn bị xử lý.

2.2.3 Các thao tác trên giao thức LDAP:

Giao thức LDAP có 9 thao tác cơ bản, có thể chia thành 3 loại:

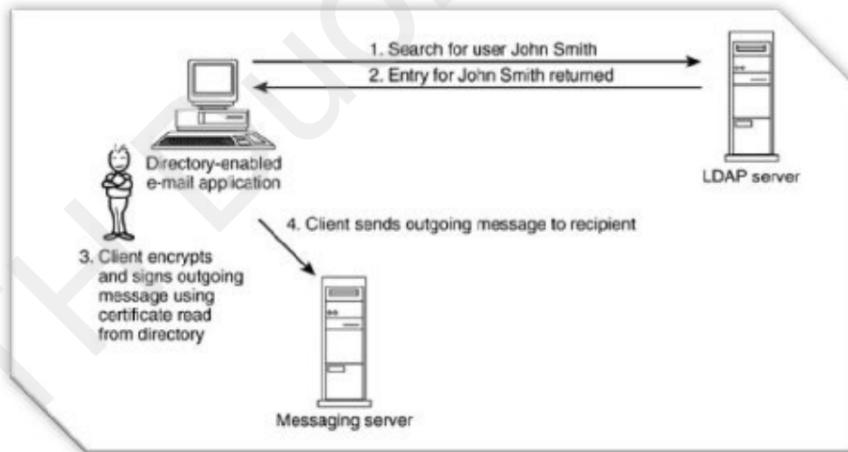
- **Thao tác truy vấn (Interrogation operations):** Search, Compare. Đây là 2 thao tác cho phép tìm kiếm và so sánh thông tin của thư mục.
- **Thao tác cập nhật (Update Operations):** Add, Delete, Modify, Modify DN (đổi tên). Những thao tác này cho phép có thể cập nhật thông tin trong thư mục.
- **Thao tác điều khiển và chứng thực:** Bind, Unbind, Abandon. Thao tác bind cho phép 1 client đến xác định bản thân nó vào thư mục bằng việc cung cấp những thông chứng thực và nhận dạng; thao tác unbind cho phép client kết thúc phiên làm việc (bind); và thao tác abandon cho phép client chỉ ra rằng client không còn quan tâm đến kết quả của thao tác mà nó đã gửi trước đó.



Hình 2.4 Một dạng trao đổi LDAP

Một dạng trao đổi giữa các client và server LDAP có thể được thực hiện như mô tả trong Hình 2. 4, trong đó các server và client LDAP thực hiện theo các bước:

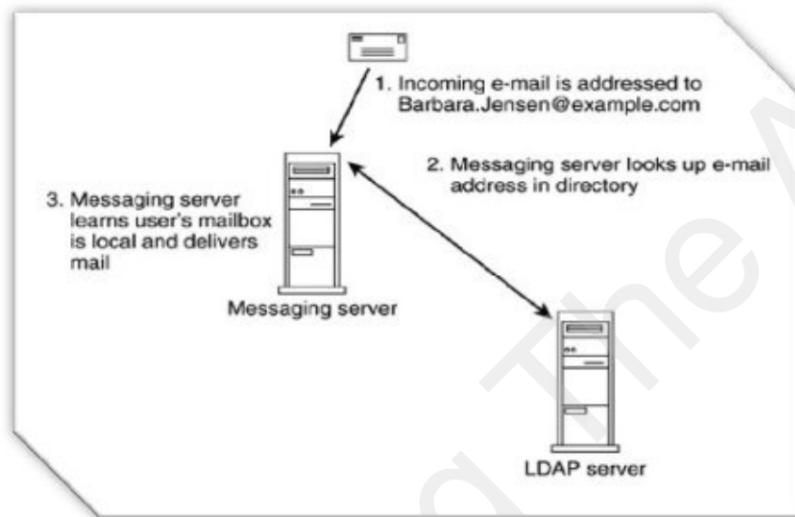
- Bước 1: Client mở một kết nối TCP đến LDAP server và gửi một thao tác yêu cầu Bind. Thao tác Bind này gồm tên của mẫu tin thư mục mà client muốn chứng thực, trong đó với những thông tin được dùng cho việc chứng thực. Những thông tin này thường là những mật khẩu đơn giản, nhưng cũng có thể chúng là những chứng chỉ số được dùng để xác thực client.
- Bước 2: Sau đó server sẽ xác nhận những thông tin Bind qua việc kiểm tra mật khẩu hay chứng chỉ số. Nếu đúng server sẽ trả về kết quả Bind thành công về cho client.
- Bước 3: Tiếp đó, client sẽ phát ra một yêu cầu tìm kiếm đến server.
- Bước 4,5: Server xử lý yêu cầu này, kết quả trong 2 mẫu tin phù hợp.
- Bước 6: Server sẽ gửi một thông điệp kết quả.
- Bước 7: Sau đó client sẽ phát một yêu cầu Unbind, báo với server là client muốn dừng kết nối.
- Bước 8: Server đóng kết nối.



Hình 2. 5 Một ứng dụng Directory-Enabled thực hiện một công việc phức tạp

Kết hợp nhiều thao tác LDAP, các client directory-enabled có thể thực hiện những thao tác phức tạp tiện ích cho người dùng. Ví dụ: 1 client thư điện tử (e-mail) như Netcape Communicator có thể tìm kiếm những người nhận thư trong thư mục, giúp người dùng đến địa chỉ thông điệp e-mail. Nó cũng có thể sử dụng một chứng chỉ số được lưu trữ trong thư mục vào các tín hiệu số và mã hóa thông điệp gửi đi. Đằng sau đó là chương

trình e-mail của người dùng thực hiện một vài các thao tác thư mục, thực hiện hiện đánh địa chỉ, ký hiệu, và mã hóa thư. Nhưng từ phía người dùng, thì những việc đó được thực hiện tự động, những việc đó được mô tả trong Hình 2. 5.



Hình 2. 6 Một ứng dụng Directory-Enabled Server

Những ứng dụng người dùng cuối không chỉ là loại ứng dụng directory-enabled. Những ứng dụng dựa trên server (Server-based) cũng thừa hưởng từ directory-enabled. Ví dụ: Server thông điệp Sun ONE có thể sử dụng một thư mục LDAP khi định tuyến thư điện tử đến, như Hình 2. 6.

2.2.4 Tính năng mở rộng của LDAP:

Ngoài 9 thao tác cơ bản, LDAPv3 được thiết kế mở rộng qua 3 phương thức:

- **Thao tác LDAP mở rộng (LDAP extended operations):** đây là những thao tác hoàn toàn mới, chúng được định nghĩa là những thao tác mở rộng LDAPv3. Nếu cần một thao tác mới, thì thao tác này có thể được định nghĩa và trở thành chuẩn mà không yêu cầu phải xây dựng lại các thành phần cốt lõi của LDAP. Ví dụ một thao tác mở rộng là StartTLS, nghĩa là báo đến server rằng client muốn sử dụng Transport Layer Security (TLS) để mã hóa và tùy chọn chứng thực khi kết nối.
- **LDAP controls:** những thành phần thông tin được kèm theo các thao tác LDAP, thay đổi hành vi của các thao tác. Ví dụ điều khiển ManageDSAIT được gửi đi với thao tác Modify khi client muốn vận dụng một số kiểu của siêu thông tin (metainformation) trong thư mục (siêu thông tin thường ẩn nén người dùng của thư mục không nhận biết được). Trong tương lai,

những control bổ sung có thể được định nghĩa nhằm thay đổi hành vi của thao tác LDAP đang hoạt động theo cách hữu ích.

- **Simple Authentication and Security Layer (SASL):** là khung hỗ trợ cho nhiều phương thức chứng thực. Nếu khung SASL được dùng để thực hiện chứng thực, thì LDAP sẽ dễ dàng được điều chỉnh để hỗ trợ những phương thức chứng thực mới và mạnh hơn. SASL cũng hỗ trợ 1 khung cho client và server để đàm phán trên những cơ chế bảo mật tàng thấp, chẳng hạn như việc mã hóa trong quá trình trao đổi giữa client và server. Mặc dù SASL không dành riêng cho LDAP, nhưng các mô hình này của SASL đều thích nghi với hầu hết các giao thức Internet.

2.2.5 Các mô hình LDAP:

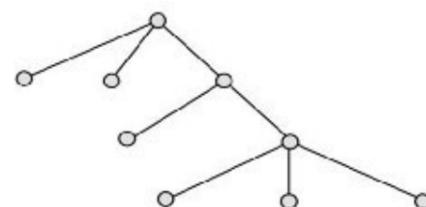
Ngoài vai trò như là một thủ tục mạng, LDAP còn định nghĩa ra bốn mô hình, các mô hình này cho phép linh động trong việc sắp đặt các thư mục:

- Mô hình LDAP information:

- Mô hình LDAP Information định nghĩa ra các kiểu của dữ liệu và các thành phần cơ bản của thông tin mà bạn có thể chứa trong thư mục. Hay chúng ta có thể nói rằng LDAP Information mô tả cách xây dựng ra các khối dữ liệu mà chúng ta có thể sử dụng để tạo ra thư mục.
- Tóm lại:
 - Thông tin trong LDAP được trình bày luận lý dưới dạng các entry (mẫu tin).
 - Các Entry thuộc về một hoặc nhiều ObjectClass (lớp đối tượng).
 - Mỗi ObjectClass được định nghĩa bởi một tập các Attribute (thuộc tính).
 - Một Attribute bao gồm kiểu thuộc tính (type) và một hay nhiều giá trị (value).
 - ObjectClass và định nghĩa kiểu thuộc tính tạo nên schema (lược đồ).

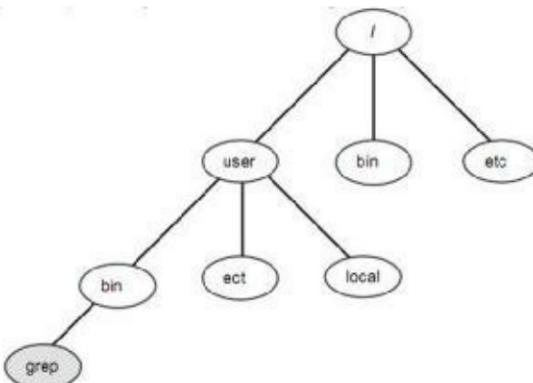
- Mô hình LDAP Naming:

- Mô hình LDAP Naming định nghĩa ra cách để chúng ta có thể sắp xếp và tham chiếu đến dữ liệu của mình. Hay chúng ta có thể



Hình 2.7 Một cây thư mục LDAP

nói rằng mô hình này mô tả cách sắp xếp các entry của chúng vào một cấu trúc luận lý, và mô hình LDAP Naming chỉ ra cách để chúng ta có thể tham chiếu đến bất kỳ một entry thư mục nào nằm trong cấu trúc đó.

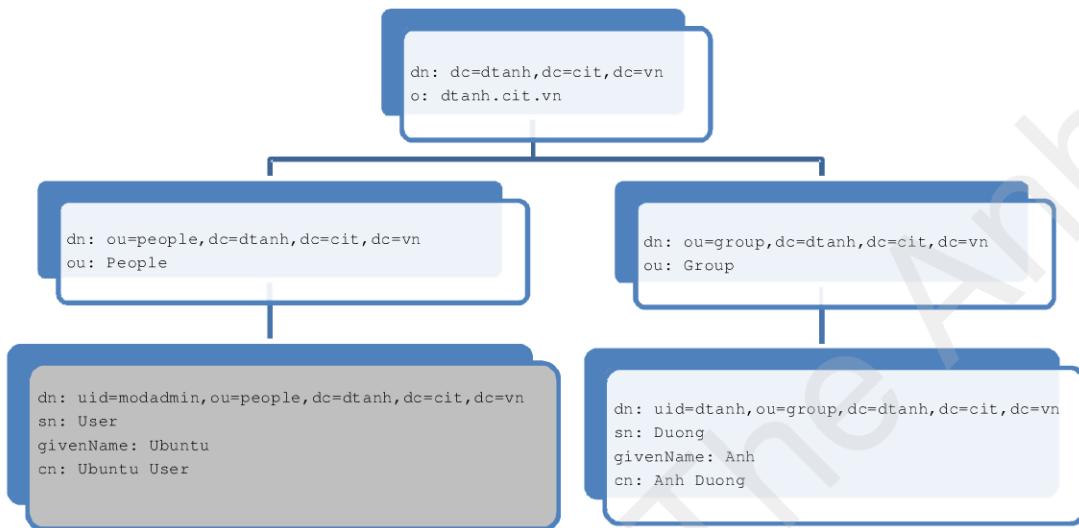


Hình 2.8 Hệ thống tập tin trong UNIX

containing all the entries that describe a person in a department, and a container containing all the group entries of a department, or a department can have its own specific entry according to the organizational structure of the department. The better the design is, the more research and effort is required.

- o Chúng ta đưa ra hệ thống tập tin UNIX để thấy được những điểm khác biệt với hệ thống thư mục LDAP, sau đó phân tích mô hình cây thư mục LDAP.
- o Có ba điểm khác biệt quan trọng :

- Điểm khác biệt đầu tiên giữa hai mô hình là trong mô hình LDAP không thực sự có một entry gốc(root). Root là nơi mà chúng ta có thể đặt các entry vào. Trên hệ thống LDAP có một entry đặc biệt được gọi là root DES chứa các thông tin về server, nhưng đây không phải là một entry thư mục bình thường.
- Khác biệt thứ hai là thư mục LDAP mỗi một node chứa dữ liệu, và cũng có thể là một container chứa các entry khác. Đây là một khác biệt với hệ thống tập tin do ở hệ thống tập tin chỉ có thư mục có thể chứa thư mục con và chỉ có tập tin mới chứa dữ liệu.



Hình 2.9 Một phần thư mục LDAP với các entry chứa thông tin

Ta có thể thấy rằng entry trong thư mục có thể đồng thời là tập tin và thư mục. Hình 2.9 minh họa khái niệm trên các entry dc=dtanh, dc=cit, dc=vn, ou=People và ou=Group tất cả đều chứa dữ liệu nhưng tất cả đều có node con cấp dưới.

- Khác biệt cuối cùng là hệ thống tập tin phân cấp và hệ thống LDAP:
 - Trong một hệ thống tập tin khi ta đi từ trái sang phải tên tập tin là cách ta thực hiện đi từ gốc(/) đến tập tin. Ví dụ như Hình 2.8 hệ thống file Unix tên file của node đậm màu là : /user/bin/grep
 - Với hệ thống thư mục LDAP tại node đậm màu có tên là uid=modadmin,ou=people,dc=dtanh,dc=cit,dc=vn nếu chúng ta đi từ trái sang phải thì chúng ta có thể quay ngược lại đỉnh của cây. Ta thấy rằng hệ thống thư mục LDAP sắp đặt có trật tự các entry của thư mục, tuy nhiên LDAP không quy định bất kỳ sự phân cấp đặt biệt nào, chúng ta có thể tự do sắp xếp hệ thống tập tin của bạn một cách có ý nghĩa nhất với bạn.

➤ Mô hình LDAP Functional:

- Phần trên chúng ta đã nói đến mô hình thông tin LDAP và cách đặt tên LDAP, bây giờ chúng ta sẽ xem xét mô hình chức năng LDAP, đây là mô hình mô tả các thao tác cho phép chúng ta có thể thao tác trên thư mục. Chúng ta sẽ khái quát hóa về mô hình chức năng LDAP.
- Mô hình chức năng LDAP chứa một tập các thao tác chia thành 3 nhóm.
 - Thao tác thẩm tra (interrogation) cho phép bạn có thể search trên thư mục và nhận dữ liệu từ thư mục.
 - Thao tác cập nhật (update): add, delete, rename và thay đổi các entry thư mục.
 - Thao tác xác thực và điều khiển(authentication and control) cho phép client xác định mình đến cho thư mục và điều khiển các hoạt động của phiên kết nối
- Với giao thức LDAP Version 3 ngoài 3 nhóm thao tác trên, còn có thao tác LDAP mở rộng, thao tác này cho phép giao thức LDAP sau này có thể mở rộng một cách có tổ chức và không làm thay đổi đến giao thức.

➤ Mô hình LDAP Security:

- Vân đề cuối cùng trong các mô hình LDAP là việc bảo vệ thông tin trong thư mục khỏi các truy cập không được phép. Khi thực hiện thao tác bind dưới một tên DN hay có thể client một người vô danh thì với mỗi user có một số quyền thao tác trên entry thư mục riêng. Và những quyền nào được entry chấp nhận tất cả những điều trên gọi là quyền điều khiển truy cập (access control). Hiện nay LDAP chưa định nghĩa ra một mô hình điều khiển truy cập, các điều khiển truy cập này được thiết lập bởi các nhà quản trị hệ thống bằng các phần mềm server.

2.2.6 Trình xử lý xác thực LDAP – CAS:

LDAP là trình xử lý xác thực mà CAS hỗ trợ, nhằm thực hiện mục đích quản lý thông tin người dùng tập trung.

Trong phần này sẽ giới thiệu về các phương thức mà CAS xác thực các thông tin:

- Đơn thuần CAS sẽ sử dụng những thông tin mà người dùng cung cấp để kết nối (bind) đến LDAP Server như một người dùng (FastBindLdapAuthenticationHandler). Phương thức này hỗ trợ một số thuộc tính (properties) sau:
 - *Filter*: thuộc tính này chính là bộ lọc trong LDAP, khi muốn thiết lập rằng người dùng có thể ở bất kỳ vị trí nào trong cây thì có thể đặt giá trị cho thuộc tính là “%u”.
 - *ignorePartialResultException*: thuộc tính này báo đê bỏ qua những ngoại lệ PartialResultExceptions được ném ra khi kết nối đến một Active Directory.
 - *contextsource*: thuộc tính này tham chiếu đến LdapContextSource (xem ở bên dưới) chứa những thiết lập để kết nối đến LDAP Server.
- Trước tiên CAS sẽ tìm kiếm người dùng trong LDAP, xem có tồn tại không. Ké đến mới có gắn kết nối (bind) đến LDAP Server như một người dùng (BindLdapAuthenticationHandler). Ngoài những thuộc tính mà phương thức đầu hỗ trợ, thì phương thức này còn thêm vào một số thuộc tính:
 - *allowMultipleAccounts*: cho phép nhiều hơn 1 tài khoản được trả về.
 - *maxNumberOfResults*: số kết quả tối đa được nhận.
 - *scope*: giá trị của thuộc tính này là một trong ba giá trị: SearchControls.OBJECT_SCOPE, SearchControls.ONELEVEL_SCOPE, hay SearchControls.SUBTREE_SCOPE.
 - *searchBase*: là nude trên cây, nơi thực hiện việc tìm kiếm.
 - *timeout*: là khoảng thời gian chờ đợi kết quả tìm kiếm được trả về.

AuthenticatedLdapContextSoure: là lớp mỏ rộng của LdapContextSource của LDAP. LdapContextSource định nghĩa một số thuộc tính quan trọng:

- *anonymousReadOnly*: có giá trị là false nếu môi trường được xác thực được tạo với thao tác chỉ dùng để đọc (read-only).
- *baseEnvironmentProperties*: nếu bất kỳ thuộc tính môi trường chỉnh sửa là cần thiết, thì nên sử dụng thuộc tính này.
- *password*: mật khẩu của người dùng chính (admin).
- *pooled*: luôn được đặt giá trị là false.

- *url* hay *urls*: url(s) đến LDAP Server, cùng với cổng giao tiếp với LDAP Server. Cổng chuẩn của LDAP là 389.
- *userDn*: tên đăng nhập của người dùng chính (admin).

2.3 Giới thiệu Liferay Portal:

2.3.1 Portal là gì?

Portal là một cổng thông tin điện tử. Khác với các trang web thông thường, portal là nơi tích hợp hầu hết các thông tin và dịch vụ cần thiết cho người dùng. Sự ra đời của portal nhằm giải quyết các nhược điểm mà các trang web thông thường mắc phải như khó mở rộng, tích hợp các dịch vụ, bảo trì, ... Đặc biệt là khả năng tùy biến khá cao của portal, cá nhân hóa, tính bảo mật khá cao.

Wikipedia (một bách khoa toàn thư được hợp tác bảo trì và chỉnh sửa trực tiếp bởi một số người dùng trên toàn thế giới) phân portal thành nhiều loại khác nhau:

- Portal cá nhân (Personal Portals): là một cổng thông tin được cá nhân tùy chỉnh để đáp ứng yêu cầu và thị hiếu của bản thân họ. Những trang portal loại này dễ dàng được chỉnh sửa và những chỉnh sửa đó được lưu trong tài khoản người dùng cá nhân.
- Portal Học Tập (Academic Portals): những địa chỉ portal học tập đáp ứng nhu cầu của các viện sĩ. Một ví dụ tiêu biểu về loại portal học tập như Uportal, được một số tổ chức giáo dục đại học tham gia cùng nhau tạo ra, dựa trên chuẩn công nghệ như JAVA, XML, JSP và J2EE. Portal học tập đều có thể tìm thấy trên mỗi quốc gia, chẳng hạn ở Việt Nam cũng đã có portal học tập của Đại Học FPT (<http://ap.fpt.edu.vn/>).
- Regional Web Portals: cung cấp thông tin cho một vị trí địa lý cụ thể. Chẳng hạn như dự báo thời tiết, bản đồ đường phố, tin tức địa phương. Ví dụ như ở Ấn Độ có một portal dạng này là Rediff (<http://www.rediff.com>), hay ở Trung Quốc thì có SINA () nó cung cấp những thông tin ở khu vực mà liên quan đến du lịch, tin tức địa phương, cổ phiếu, ...
- Government Web Portals: nhiều chính phủ trên toàn thế giới cung cấp một portal cho công dân của họ. Chẳng hạn như cổng thông tin điện tử quốc gia của Việt Nam (<http://www.chinhphu.vn>), cung cấp thông tin hữu ích liên quan đến chính phủ cho các công dân Việt Nam.
- Corporate Web Portals: dạng này còn được gọi là portal mạng nội bộ (intranet portals). Các mạng nội bộ của công ty cung cấp một cái nhìn về công ty một cách hợp nhất đến các nhân viên và cho phép họ cá nhân hóa,

tùy chỉnh hiển thị nội dung trang web, cho phép tạo và công bố tiến trình công việc để tạo điều kiện thuận lợi cho sự hợp tác giữa các bộ phận trong công ty. Bên cạnh đó, các người dùng có thể tạo và chia sẻ kiến thức trên các trang wiki, giúp cho việc kinh doanh của công ty. Ngoài ra, portal mang cục bộ mở rộng ra cho người dùng internet truy cập vào một số thông tin của công ty hay còn được gọi là extranet portal. Khi đó portal của công ty có thể lưu trữ trên máy chủ nội bộ hoặc máy chủ bên ngoài tùy thuộc vào khả năng của công ty.

- Domain-Specific Portals: portal hướng đến ngành công nghiệp cá biệt (particular industry) thì được gọi là domain-specific portals. Ví dụ, một portal cho những người kinh doanh bất động sản mang lại một đại lý khu vực trên một trang web đơn lẻ và cho phép các khách hàng đến để mua bán tài sản của họ. Một người kinh doanh bất động sản vẫn có thể tạo một portal nhằm tạo điều kiện thuận lợi cho việc mua bán bất động sản.
- Portal Thể Thao (Sports Portals): một vài portal phục vụ cho nhu cầu của những người yêu thể thao. Một ví dụ điển hình của một portal thể thao là ESPN STAR Sport (<http://www.espnstar.com/>) bao gồm các môn như bóng đá, quần vợt, cricket quốc tế, motorsport, khúc quân cầm, golf. Bên cạnh đó cũng có một số portal thể thao khá phổ biến như Sky Sports (<http://www.skysports.com/>), Sportal (<http://sportal.nic.in/>), và Sify Sports (<http://sify.com/sports/>). Hầu như các portal thể thao đều bao trùm nhiều khía cạnh khác nhau của các môn thể thao như trực tiếp điểm số, trực tiếp trận đấu, tường thuật lại các trận đấu, phân tích các trận đấu, ...

2.3.2 Thuận lợi của Portal

Qua phần phân loại, thì chúng ta chủ yếu đề cập về mặt tập hợp nội dung và tùy chỉnh trang bố trí (page-layout), nhưng cũng có đề cập về tính năng mở rộng mà portal có thể cung cấp: sự hợp tác giữa các cộng đồng người dùng. Thực sự, một người thiết kế portal có thể tạo ra những cộng đồng người dùng khác nhau trên cùng một portal và yêu cầu những người dùng đăng ký với mỗi cộng đồng đó. Nhà thiết kế có thể trình bày nội dung khác nhau, giao diện khác nhau, và các tính năng hợp tác khác nhau mà phụ thuộc vào cộng đồng người dùng tham gia. Những tính năng hợp tác mà có thể tìm thấy trong mỗi portal là:

- ✓ Thảo luận nhóm (Group Discussions): các cuộc thảo luận nhóm được lưu trên portal, người dùng có thể bàn luận về một chủ đề đã được thiết lập trên một bảng thông điệp. Chức năng hợp tác này cung cấp một vài thuận lợi

hơn so với nhóm thảo luận thông thường qua e-mail. Người dùng có thể tham gia hoặc rời khỏi cuộc thảo luận bất kỳ lúc nào mà không bị ảnh hưởng gì đến những người khác, và vì đã được lưu trữ nên người dùng có thể xem lại cuộc thảo luận.

- ✓ Blogs: một portal cho phép người dùng đưa ra những quan điểm trên blog của họ, giống như những người dùng trong các trang blog thuần như BLOGGER (<http://www.blogger.com>) hay Windows Live Spaces (<http://home.spaces.live.com/>). Blog có thể công bố đến những người dùng khác các quan điểm, mà các quan điểm này được người dùng có kinh nghiệm chuyên sâu đưa ra, chẳng hạn như về phương hướng thị trường và công nghiệp.
- ✓ Chia sẻ tài liệu (Document Sharing): một portal cho phép chia sẻ những tài liệu đang tồn tại và các phương tiện khác như ảnh chụp. Chức năng này yêu cầu quản lý đúng cách.
- ✓ Wikis: trang portal có thể lưu trữ trang wiki, trang này cho phép người dùng tạo ra các trang web, chỉnh sửa chúng, liên kết chúng lại với nhau. Chức năng này giống như chia sẻ quyền sở hữu, dù người dùng đó ở đâu cũng có thể chia sẻ các ý kiến với một tinh thần cộng tác cao.
- ✓ Lịch được chia sẻ (shared calendars): portal lưu trữ các lịch chia sẻ, cho thấy sự tiện dụng trong việc quản lý các sự kiện của công ty hay của người dùng. Chức năng này giúp cho người dùng sắp xếp những cuộc gặp gỡ, hội họp và gửi lời mời.

2.3.3 Tạo một portal với Liferay:

Tất cả những điều trên đã giới thiệu được phần nào về khái niệm cũng như lợi ích mà portal mang lại so với trang web bình thường. Một portal chứa nhiều ứng dụng khác nhau mà có thể có hoặc không liên kết chúng lại với nhau, chẳng hạn như blog, ứng dụng quản lý tài liệu, wiki, lịch, ... nhưng rõ ràng, chẳng có một nhà cung cấp nào có thể cung cấp đầy đủ tất cả các công cụ để làm nên một portal có đầy đủ các tính năng đặc trưng đã nêu. Điều này sẽ dẫn đến việc sử dụng các công cụ có tính liên kết chặt chẽ của nhiều nhà cung cấp khác nhau, nên có thể không thống nhất công nghệ sử dụng, vấn đề thách thức đặt ra là tích hợp chúng lại bởi vì các ứng dụng của portal cần phải có khả năng làm việc được với nhau.

Nhiều nhà cung cấp cũng đã cung cấp các công cụ để tạo ra portal và máy chủ lưu trữ trang portal. Một số công cụ như Oracle WebLogic Portal, IBM WebSphere Portal

Server, Sun Java Portal Server (hay GlassFish Web Space Server), và Microsoft Office SharePoint Server. Với công nghệ mã nguồn mở, thì Liferay là máy chủ portal phổ biến.

Về cơ bản portal Liferay cung cấp khung sườn của nhiều dạng portal như đã đề cập trước đó. Đơn giản liferay cũng giống như một ứng dụng web được lưu trữ trên máy chủ web. Liferay hỗ trợ nhiều dạng máy chủ do được biên dịch hoàn toàn theo chuẩn.

Liferay cung cấp môi trường phát triển toàn diện để người dùng có thể sử dụng tạo trang portal. Liferay cũng cung cấp một môi trường thực thi lưu trữ cho các ứng dụng portal trên nền Java, còn được gọi là portlet. Liferay sẽ có một chỗ để chứa những portlet, được thiết lập, tùy chỉnh hiển thị.



Hình 3.1 Một portal sử dụng Liferay

Hình 1.1 là một trang portal điển hình chạy bằng Liferay, nhìn bề ngoài portal Liferay cũng giống như các trang portal khác.

Để nhận biết sự tiện lợi của Liferay, thì có một số tính năng cơ bản cần quan tâm:

- **Dễ sử dụng:** người dùng có thể thêm những ứng dụng khác nhau trên trang portal bằng cách sử dụng tính năng kéo thả (drag-and-drop), hay di chuyển vị trí của các ứng dụng với việc nhấp (clicking) và kéo (dragging). Nếu muốn xóa một ứng dụng đang hiển thị trên trang portal thì chỉ việc click chuột vào biểu tượng đóng ứng dụng. Và cũng dễ dàng cho việc thay đổi

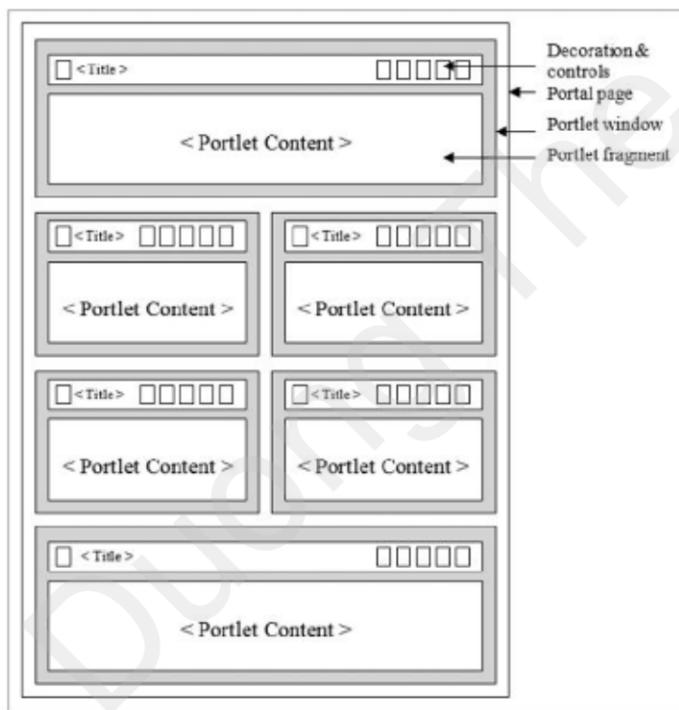
cách hiển thị (look and feel) của trang portal là thay đổi giữa các chủ đề (theme) do liferay hoặc bên thứ ba cung cấp. Chính vì thế, người dùng dễ dàng tạo trang portal với liferay và cấu hình theo ý riêng của từng người mà không cần viết bất kỳ một dòng mã chương trình nào.

- Hỗ trợ nhiều ứng dụng: Liferay cung cấp một kho phong phú các ứng dụng (hay portlet) cho người dùng, kể cả wiki, blog, tán gẫu (chat), và diễn đàn thảo luận. Ngoài ra, người dùng còn có thể sử dụng những ứng dụng sáp nhập được cộng đồng người dùng tạo sẵn.
- Cấp giấy phép tự do và là mã nguồn mở: việc sử dụng liferay trên máy chủ riêng là hoàn toàn miễn phí. Công ty có một chính sách cấp phép tự do: người dùng không cần bất kỳ giấy phép nào khi sử dụng sản phẩm, ngay cả khi vì mục đích thương mại.(Liferay chỉ có doanh thu qua việc bán hỗ trợ và đào tạo).Thêm vào đó, Liferay cũng đã làm toàn bộ mã nguồn portal sẵn dùng cho cộng đồng phát triển. Ý nghĩa của việc chỉnh sửa mã nguồn:
 - Người dùng có thể thêm tính năng cho portal của mình qua việc chỉnh sửa mã chương trình.
 - Góp phần cải thiện cho Liferay.
 - Đội ngũ phát triển viên có thể thêm tính năng cho Liferay qua việc tạo ra các plugin.
- Extensibility: là khả năng dễ dàng mở rộng thêm tính năng mới mà không cần bất kỳ một thay đổi nào về mã nguồn. Điều này giúp cho tính linh động trong chuyển đổi sang phiên bản mới của Liferay khi được công bố.
- Scalability: Liferay có khả năng xử lý một cơ sở người dùng lớn.
- Sự hỗ trợ đa ngôn ngữ: Liferay cung cấp sự hỗ trợ tốt cho việc quốc tế hóa portal. Khi portal có hệ thống người dùng không sử dụng tiếng Anh tương đối lớn thì khi đó việc tính hợp thêm ngôn ngữ mới cũng dễ dàng, đơn giản chỉ sử dụng những gói tài nguyên thích hợp với ngôn ngữ cần dùng.
- Tích hợp với các công cụ khác: Liferay dễ dàng tích hợp thêm công cụ của bên thứ ba. Chẳng hạn, Liferay có hệ quản trị nội dung (Content Management System – CMS) riêng, nhưng cũng hỗ trợ công cụ mạnh hơn Alfresco (<http://www.alfresco.com/>). Liferay cũng hỗ trợ máy chủ Lightweight Directory Access Protocol (LDAP), và các hệ quản trị cơ sở dữ liệu phổ biến như: Oracle, IBM DB2, MySQL, SQL Server, Informix...
- Gắn chặt với các chuẩn công nghiệp: Liferay dựa trên các chuẩn công nghệ nên dễ tích hợp với chuẩn công nghệ khác. Sau đây là một số chuẩn công nghệ có thể tích hợp với Liferay:

- PHP: là một ngôn ngữ kịch bản mục đích chung, đặc biệt thích hợp cho việc phát triển web, có thể nhúng vào trong HTML.
- Ruby: một ngôn ngữ lập trình nguồn mở, động.

Liferay được tạo hoàn toàn với Java, một ngôn ngữ lập trình chuẩn. Chính vì thế, người dùng có thể dễ dàng chỉnh sửa, bảo trì, mở rộng Liferay.

2.3.4 Cấu trúc bên trong trang:



Hình 3.2 Cấu trúc bên trong một trang portal

Hình 1.2 cho thấy một trang portal khởi động một tập portlet. Mỗi portlet có mỗi cách trang trí và điều khiển riêng, điều khiển ở đây là tùy biến với portlet tương ứng. Ví dụ như, dễ dàng thay đổi giao diện (look and feel) của portlet qua màu nền, màu khung, chiều rộng, kiểu chữ,...

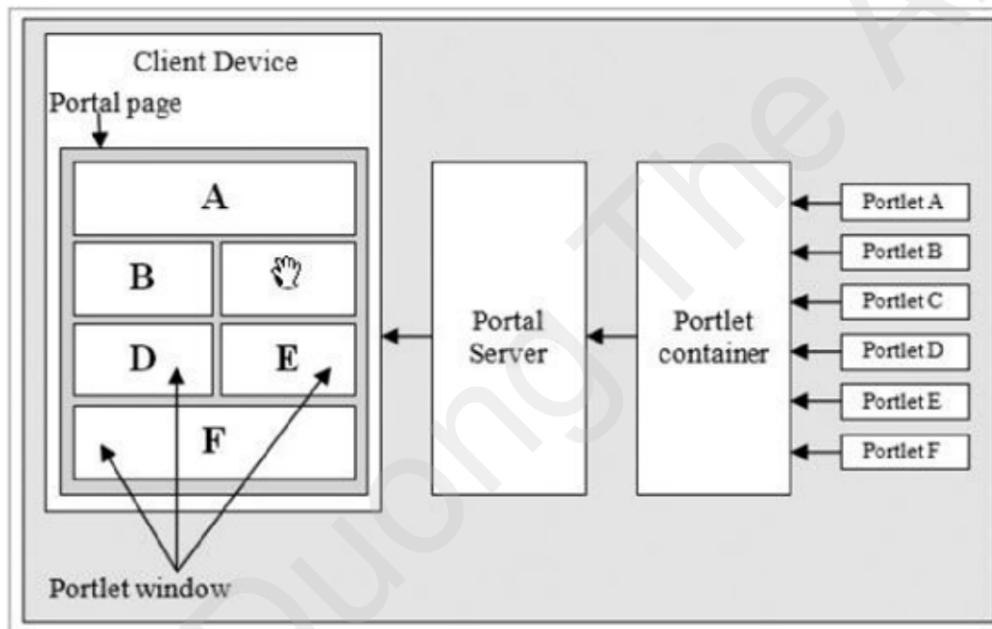
Với việc cấu hình điều khiển, có thể đặt quyền cho mỗi portlet riêng lẻ, do đó chỉ những người dùng được chỉ định mới có quyền sử dụng.

Người dùng có thể tùy biến vị trí hiển thị của portlet trên trang bằng cách ứng dụng một mẫu thiết kế (layout), hoặc có thể chỉnh sửa thiết kế theo ý riêng và ứng dụng

vào trang portal của mình, ngoài ra cũng có thể thêm hoặc xóa portlet trên trang web bất cứ lúc nào.

Các portlet khác nhau có thể giao tiếp với nhau trên cùng một trang, mà không cần quan tâm đến công nghệ riêng của từng portlet. Kho portlet chịu trách nhiệm cung cấp tất cả các tính năng trên.

2.3.5 Tiến trình tạo trang:



Hình 3.3 Một trang portal được sinh ra như thế nào

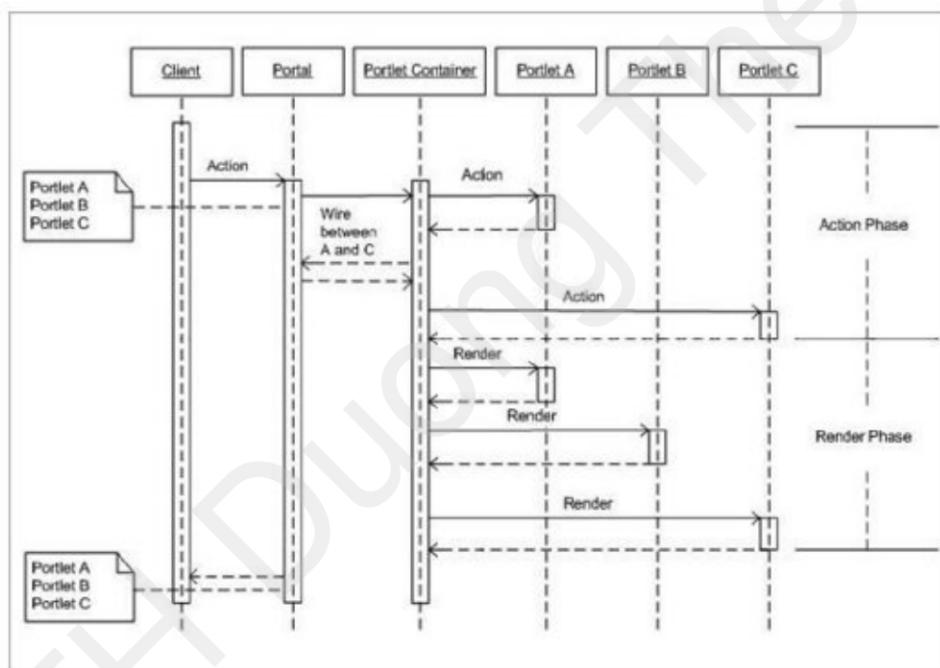
Giả sử trang portal được nêu ra với 6 portlet được nhúng trong trang portal (như hình 1.3). Sau đây là xử lý diễn ra trong quá trình sinh ra trang portal:

1. Mỗi portlet (A,B,C,D,E,F) tạo nên mỗi nội dung cho người dùng. Các nội dung có thể được tạo động hoặc tĩnh, phụ thuộc vào tính luận lý của ứng dụng portlet. Chú ý, mỗi portlet cơ bản là một ứng dụng Java nên sẽ chạy nhiều mã khi hoạt động.
2. Kho (container) nhận nội dung được sinh ra bởi các portlet.
3. Container truyền các nội dung đến máy chủ portal.
4. Máy chủ portal tạo trang portal, về cơ bản là một chuỗi các mã HTML mà trình duyệt có thể hiểu. Trong quá trình tạo trang, máy chủ portal ứng dụng cách bố trí trang được thiết kế để đặt mỗi portlet đến vị trí thích hợp.

5. Máy chủ gửi trang vừa được tạo đến thiết bị khách (trong trường hợp này là trình duyệt).
6. Trình duyệt hiển thị nội dung lên cho người dùng.

Trong ngữ cảnh này, portal sẵn sang tương tác nhiều hơn với người dùng. Người dùng nhìn vào nội dung được trình bày và có thể quyết định tìm hiểu thêm thông tin khi nhấp vào 1 trong những biểu tượng được hiển thị trên trang. Khi người dùng yêu cầu thêm thông tin thì xem xét, hành động sẽ lấy vị trí nào trong máy chủ portal và kho portlet.

2.3.6 Cách giải quyết yêu cầu:



Hình 3.4 Sơ đồ trình tự về yêu cầu người dùng

Giả sử portal chứa 3 portlet (A,B,C). Người dùng bắt đầu một hành động trên trang portal, yêu cầu nội dung của portlet A, B và C để được bô nghĩa. Sự kiện lấy vị trí trong suốt quá trình tương tác đã được miêu tả trong mô hình trình tự hình 1.4:

1. Người dùng yêu cầu một dữ liệu cập nhật khi click vào trang portal.
2. Yêu cầu người dùng đã sinh ra một sự kiện hành động trên máy chủ portal.
3. Máy chủ portal sinh ra một sự kiện trên kho portlet.

4. Kho portlet xác định yêu cầu người dùng yêu cầu dữ liệu ở portlet A và C cần được cập nhật.
5. Kho làm ra một yêu cầu ACTION gửi đến portlet A để thực hiện dữ liệu cập nhật.
6. Portlet A thực hiện hành động được yêu cầu và trả về kết quả, nhiều trường hợp trả về kho.
7. Kho làm một yêu cầu ACTION gửi đến portlet C.
8. Portlet C thực hiện hành động được yêu cầu và gửi kết quả, nhiều trường hợp trả về kho.
9. Đến lúc này, yêu cầu ACTION của người dùng đã hoàn thành. Container đã có dữ liệu cần thiết để hồi âm lại cho người dùng.
10. Tiến trình hồi âm lại cho yêu cầu người dùng bắt đầu.
11. Container khởi động sự kiện RENDER đến cả 3 portlet (có thể theo thứ tự hoặc song song).
12. Container tập hợp các hồi đáp của 3 portlet.
13. Container trả về trang được cập nhật cho máy chủ.
14. Máy chủ hiển thị trang được cập nhật trên trình duyệt của người dùng.
15. Đến đây, yêu cầu đã được xử lý hoàn tất.
16. Nay, máy chủ đang trong trạng thái chờ một sự tương tác khác từ người dùng.

2.3.7 Quản lý nội dung với CMS:

➤ Khái niệm CMS:

Content Management System là cụm từ được viết đầy đủ của CMS được gọi là hệ quản trị nội dung hay hệ thống quản trị nội dung. Đây là phần mềm để tổ chức và tạo ra môi trường thuận lợi nhằm mục đích xây dựng hệ thống tài liệu và các loại nội dung khác một cách thống nhất.

➤ CMS trong Liferay:

Liferay Portal cung cấp hai chức năng chính trong CMS: Quản lý nội dung và xuất bản các nội dung đó.

Quản lý dữ liệu: Việc quản lý dữ liệu được cung cấp qua các portlet Document Library, Image Gallery. Document Library để quản lý các tập tin. Còn Image Gallery dùng để quản lý các hình ảnh trong hệ thống.

Xuất bản nội dung: Web Content Display là porlet cho phép người dùng tạo, chỉnh sửa và xuất bản các nội dung cần hiển thị. Trong porlet này ta có thể tạo ra các mẫu template có sẵn để định dạng cho nội dung cần hiển thị, cho phép nội dung được tạo được phép tìm kiếm với porlet Web Content Search hay không, nhóm nội dung vào các Tags hay Category để dễ dàng cho việc xuất bản với porlet Assert Publisher. Porlet Assert Publisher là công cụ cho phép ta hiển thị đồng loạt các nội dung trong porlet Web Content Display theo Tags. Nội dung trong porlet Assert Publisher được hiển thị một cách linh động và tự động cập nhật nếu được thiết lập.

Ngoài ra Liferay CMS còn có các chức năng sau:

- Upload được nhiều định dạng tập tin vào hệ thống. Hỗ trợ 22 ngôn ngữ và cho phép định nghĩa thêm các ngôn ngữ khác.
- Tìm kiếm nội dung trong hệ thống nhanh chóng qua porlet Web Content Search.
- Với porlet Navigation ta sẽ nhận được cây thư mục liên kết đến các trang của portal trong cùng một Community hay Organization. SiteMap là porlet có khả năng hiển thị liên kết linh động hơn nhiều so với Navigation.
- Việc hiệu chỉnh style của các porlet kể trên hết sức dễ dàng qua công cụ *Look and Feel* tương ứng với từng porlet hay từng loại porlet.

2.4 Moodle:

2.4.1 Moodle là gì?

Moodle là một hệ thống quản lý học tập (Learning Management System - LMS hoặc người ta còn gọi là Course Management System hoặc VLE - Virtual Learning Environment) mã nguồn mở (do đó miễn phí và có thể chỉnh sửa được mã nguồn), cho phép tạo các khóa học trên mạng Internet hay các website học tập trực tuyến.

Moodle (viết tắt của Modular Object-Oriented Dynamic Learning Environment) được sáng lập năm 1999 bởi Martin Dougiamas, người tiếp tục điều hành và phát triển chính của dự án. Do không hài lòng với hệ thống LMS/LCMS thương mại WebCT trong trường học Curtin của Úc, Martin đã quyết tâm xây dựng một hệ thống LMS mã nguồn mở hướng tới giáo dục và người dùng hơn. Từ đó đến nay Moodle có sự phát triển vượt bậc và thu hút được sự quan tâm của hầu hết các quốc gia trên thế giới và ngay cả những công ty bán LMS/LCMS thương mại lớn nhất như BlackCT (BlackBoard + WebCT) cũng có các chiến lược riêng để cạnh tranh với Moodle.

Moodle nổi bật là thiết kế hướng tới giáo dục (<http://docs.moodle.org/en/Philosophy>), dành cho những người làm trong lĩnh vực giáo dục.

Moodle rất dễ dùng với giao diện trực quan, giáo viên chỉ mất một thời gian ngắn để làm quen và có thể sử dụng thành thạo.

Do thiết kế dựa module nên Moodle cho phép bạn chỉnh sửa giao diện bằng cách dùng các theme có trước (<http://moodle.org/mod/data/view.php?id=6552>) hoặc tạo thêm một theme (<http://docs.moodle.org/en/Theme>) mới cho riêng mình.

Tài liệu hỗ trợ (<http://docs.moodle.org/>) của Moodle rất đồ sộ và chi tiết, khác hẳn với nhiều dự án mã nguồn mở khác.

Moodle phù hợp với nhiều cấp học và hình thức đào tạo: phổ thông, đại học/cao đẳng, không chính quy, trong các tổ chức/công ty.

Moodle rất đáng tin cậy, có trên 10.000 site (trên thống kê tại moodle.org). Thế giới đã dùng Moodle tại 160 quốc gia và đã được dịch ra 75 ngôn ngữ khác nhau. Có trên 100 nghìn (<http://moodle.org/stats/>) người đã đăng ký tham gia cộng đồng Moodle (moodle.org) và sẵn sàng giúp bạn giải quyết khó khăn. Nếu bạn cần sự giúp đỡ chuyên nghiệp về cài đặt, hosting, tư vấn sử dụng Moodle, phát triển thêm các tính năng mới, và tích hợp Moodle với các hệ thống đã có trong trường của bạn, bạn có thể chọn cho mình một trong các công ty Moodle Partners (<http://moodle.com/partners/list/>) (Khoảng 30 công ty).

Moodle phát triển dựa trên PHP (Ngôn ngữ được dùng bởi các công ty Web lớn như Yahoo, Flickr, Baidu, Digg, CNET) có thể mở rộng từ một lớp học nhỏ đến các trường đại học lớn (http://docs.moodle.org/en/Installations_10000_plus) trên 50 000 sinh viên (ví dụ đại học Open Polytechnic của Newzealand hoặc sắp tới đây là đại học mở Anh - Open University of UK, trường đại học cung cấp đào tạo từ xa lớn nhất châu Âu, và đại học mở Canada, Athabasca University). Bạn có thể dùng Moodle với các database (<http://moodle.org/mod/data/view.php?d=13&rid=22>) mã nguồn mở như MySQL hoặc PostgreSQL. Từ phiên bản 1.7, sẽ hỗ trợ thêm các database thương mại như Oracle, Microsoft SQL để các bạn có thêm nhiều cơ hội lựa chọn.

Để biết mọi người nghĩ gì về Moodle, các nghiên cứu về Moodle, cũng như so sánh Moodle với các hệ thống khác, bạn đọc tiếp tại: <http://moodle.org/buzz/>. Về tương lai phát triển của Moodle, bạn xem tại: <http://docs.moodle.org/en/Roadmap>.

Cộng đồng Moodle Việt Nam được thành lập tháng 3 năm 2005 với mục đích xây dựng phiên bản tiếng Việt và hỗ trợ các trường triển khai Moodle. Từ đó đến nay, nhiều trường đại học, tổ chức và cá nhân ở Việt Nam đã dùng Moodle. Có thể nói Moodle là một trong các LMS thông dụng nhất tại Việt Nam. Cộng đồng Moodle Việt Nam giúp bạn giải quyết các khó khăn về cài đặt, cách dùng các tính năng, cũng như cách chỉnh sửa và phát triển. Nhớ rằng cộng đồng Moodle Việt Nam được xây dựng bằng chính Moodle.

2.4.2 Tại sao dùng Moodle?

Câu hỏi này được đặt ra khi nhiều trường đại học đã dùng BlackBoard hoặc WebCT (do họ đã sát nhập nên chúng ta gọi là BlackCT) chuyển sang dùng Moodle. Với những ai không biết thì BlackBoard và WebCT là hai LMS/LCMS ra đời sớm và chiếm thị phần lớn nhất trên thế giới trong số các hệ thống thương mại. Một thống kê tham khảo về thị phần các LMS/LCMS chính Moodle+BlackCT+Sakai có tại <http://www.zacker.org/higher-ed-lms-market-penetration-moodle-vs-blackboard-vs-sakai>.

Dưới đây là các lí do chính (Dựa trên nghiên cứu của Terence Armentano được đưa lên tại địa chỉ: <http://moodle.org/mod/forum/discuss.php?d=35845>)

- *Phần mềm nguồn mở giúp trường đại học của bạn không phụ thuộc vào một công ty phần mềm đóng.*
 - Ví dụ 1 – LMS (Learning Management System) đóng có thể ảnh hưởng rất sâu đến một trường đại học cho đến mức mà không thể quay lại. Giáo viên quá quen với nó. Sinh viên và các nhân viên khác cũng vậy. Đến lúc này công ty bán LMS nhận ra sự phụ thuộc của bạn vào sản phẩm này và bắt đầu tăng giá, hỗ trợ ít hơn, bắt khách hàng mua các sản phẩm bổ sung và khách hàng bắt buộc phải làm theo, không còn sự lựa chọn nào khác.
 - Ví dụ 2 – Nếu cần hỗ trợ, khách hàng phải dựa vào công ty bán sản phẩm để nâng cấp và chỉnh sửa vì khách hàng không thể có mã nguồn trong tay. Với mã nguồn mở, người dùng có thể tự sửa hoặc trả cho các công ty khác hỗ trợ, thường thì rẻ hơn vì người dùng có thể chọn được nhiều công ty. Hơn nữa, nếu không hài lòng với một công ty, người dùng có thể tìm các công ty khác để hỗ trợ. Moodle có khoảng 30 công ty có thể hỗ trợ người dùng. Hơn nữa, nếu có những chuyên gia tin học tốt tại nội bộ thì không cần thuê bên ngoài.
- *Tùy biến được (Customizable):* Moodle có thể tùy biến và cấu hình mềm dẻo một cách ngạc nhiên. Mã nguồn mở được đưa ra công khai do đó

người dùng có thể tùy biến hệ thống để phù hợp với các yêu cầu đào tạo và thuê lập trình viên làm chuyện đó thay cho mình.

- Ví dụ: nếu trường đại học muốn xây dựng một module XYZ thì họ có thể tự phát triển bên trong hoặc gửi yêu cầu đó lên cộng đồng mã nguồn mở và một người lập trình viên có thể xây dựng module đó miễn phí. Ngay cả khi bạn không phải là một lập trình viên, bạn vẫn có thể cài đặt Moodle trên một server, tạo các khóa học, và cài thêm các module bổ sung, và gỡ các rắc rối với sự trợ giúp của cộng đồng Moodle.
- *Hỗ trợ*: Các mức độ hỗ trợ cho một phần mềm mã nguồn mở tốt thật đáng kinh ngạc. Cộng đồng, nhân viên công nghệ thông tin có sẵn, hoặc các công ty bên ngoài là các lựa chọn cho người dùng.
- *Chất lượng*: Đôi khi phần mềm mã nguồn mở, như trong trường hợp của Moodle và Sakai, bằng hoặc tốt hơn Blackboard /WebCT trong các khía cạnh. Bởi cộng đồng các nhà giáo dục, chuyên gia máy tính, và các chuyên gia thiết kế giảng dạy chính là những người phát triển Moodle, và kết quả là bạn có trong tay một sản phẩm đáp ứng tốt các yêu cầu người dùng.
 - Ví dụ, Moodle có các tính năng hướng tới giáo dục vì chúng được xây dựng bởi những người làm trong lĩnh vực giáo dục.
- *Moodle được hỗ trợ tích cực bởi những người làm trong lĩnh vực giáo dục*: Họ là những người có trình độ công nghệ thông tin tốt và có kinh nghiệm trong giảng dạy. Họ chính là những người dùng LMS và có thể hỗ trợ bạn.
- *Sự tự do*: Bạn có nhiều sự lựa chọn hơn và không bao giờ có cảm giác là ‘nô lệ’ của phần mềm.
- *Ảnh hưởng trên toàn thế giới*: Bởi vì Moodle có một cộng đồng lớn như vậy, phần mềm được dịch ra hơn 75 ngôn ngữ và được sử dụng tại 160 quốc gia khác nhau. Người dùng rất ít khi tìm được một phần mềm đóng thông dụng được dịch ra hơn 10 ngôn ngữ khác nhau.
- *Moodle, giống như các công nghệ mã nguồn mở khác, có thể tải về và sử dụng miễn phí*: Mã nguồn mở dùng mô hình kinh doanh khác với mô hình mà người dùng từng biết.
 - Ví dụ, bạn có thể mở một công ty tư vấn Moodle và thuê một lập trình viên để phát triển phần mềm và chia sẻ nó miễn phí cho cộng đồng bởi vì càng có nhiều người dùng nó công ty của bạn càng có cơ hội kinh doanh.
- *Cơ hội cho các sinh viên tham gia dự án*: Thật là tốt khi bạn tạo điều kiện cho các sinh viên khoa học máy tính (công nghệ thông tin) có cơ hội để

phát triển một module cho LMS Moodle. Sinh viên có thể xây dựng module cho LMS Moodle và chia sẻ nó cho cộng đồng toàn cầu. Nếu module đủ tốt, nó sẽ được tích hợp vào phiên bản mới Moodle thường được phát hành 6 tháng một lần. Bởi vì Moodle thiết kế dựa trên module, xây dựng module mới cho Moodle khá đơn giản nếu bạn biết PHP. (Ví dụ như sinh viên *Phạm Minh Đức* - Đại học BK Hà Nội đã phát triển thành công module SCORM (<http://moodle.org/mod/data/view.php?d=13&rid=329>) 2004, sau đó đóng góp cho cộng đồng Moodle).

- *Với mô hình mở như Moodle, cho phép bạn trao đổi trực tiếp với chính những người phát triển phần mềm, góp ý kiến và yêu cầu chỉnh sửa:* Nếu muốn bạn có thể nghe cuộc phỏng vấn với *Martin Dougiamous*, người sáng lập Moodle và hiện tại vẫn đang là người điều hành chính Moodle, về tương lai của Moodle tại <http://technosavvy.org/?p=329>.

2.4.3 Sử dụng Moodle:

Nếu cần biết thêm về cách sử dụng Moodle một cách cận kề hơn. Các đọc giả có thể tải quyển ebook Using Moodle tại địa chỉ http://download.moodle.org/download.php/docs/en/using_moodle_2e.zip.

2.5 Servlet:

2.5.1 Servlet là gì?

Servlet là module viết bằng JAVA chạy với ứng dụng server để trả lời cho các yêu cầu của client. Servlet không gắn với giao thức client-server đặc trưng nhưng thường được sử dụng với HTTP và từ “Servlet” thường được dùng với nghĩa là HTTP Servlet.

Servlet sử dụng các lớp mở rộng theo chuẩn Java trong các gói (package) javax.servlet (Servlet Framework cơ bản) và javax.servlet.http (phần mở rộng của Servlet Framework cho Servlet trả lời các yêu cầu HTTP). Từ khi Servlet được viết bằng ngôn ngữ Java có tính thích nghi cao và tuân theo chuẩn framework, thì servlet cung cấp một phương tiện để tạo các phần mở rộng cho máy chủ phức tạp tại server hay hệ điều hành độc lập.

HTTP Servlet có một số lợi ích sau:

- ✓ Xử lý, lưu trữ dữ liệu được gửi từ một form HTML.
- ✓ Cung cấp nội dung động. Ví dụ: trả về kết quả của một câu truy vấn cơ sở dữ liệu cho client.

- ✓ Quản lý thông tin trang thái trên đầu trang HTTP. Ví dụ: một hệ thống giỏ hàng mua sắm trực tuyến, trong đó quản lý giỏ hàng cho nhiều khách đồng thời và kết nối đúng yêu cầu với khách hàng.

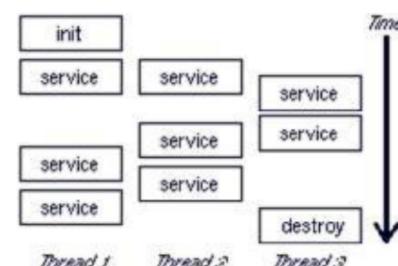
2.5.2 Kiến trúc Servlet cơ bản:

Hình thức chung của một Servlet là một thể hiện của một lớp (class) mà thực hiện giao diện javax.servlet.Servlet. Tuy nhiên hầu hết servlet mở rộng một trong những chuẩn triển khai thực hiện giao diện đó, cụ thể là javax.servlet.GenericServlet và javax.servlet.http.HttpServlet. Ở đây, chúng ta chỉ quan tâm đến HTTP Servlet, lớp mở rộng javax.servlet.http.HttpServlet.

Để khởi tạo một Servlet, một ứng dụng server tải về các lớp Servlet (và các lớp mà Servlet tham chiếu đến). Tạo một thể hiện bằng việc gọi một phương thức xây dựng không đối số. Kế đến là gọi phương thức init(ServletConfig config) của Servlet. Servlet nên thực hiện thiết lập các thủ tục (procedure) một lần ở bên trong phương thức này và lưu trữ đối tượng ServletConfig, do nó có thể sẽ được gọi ngay sau đó qua phương thức getServletConfig() của Servlet. Điều này được xử lý bởi GenericServlet. Servlet mở rộng GenericServlet (hay lớp con HttpServlet) nên gọi super.init(config) vào đầu phương thức init để sử dụng tính năng này. Đối tượng ServletConfig chứa các tham số Servlet và tham chiếu đến ServletContext của Servlet. Phương thức init được đảm bảo chỉ gọi duy nhất một lần trong một vòng đời (lifecycle) Servlet. Ở đây cũng không cần là an toàn luồng (thread-safe) bởi vì phương thức service chỉ được gọi khi nào init trả về.

Khi Servlet được khởi tạo, phương thức service(ServletRequest req, ServletResponse res) được gọi cho mỗi yêu cầu đến Servlet. Phương thức này có thể được gọi đồng thời (tức là đa luồng có thể gọi phương thức này tại cùng một thời điểm) do đó phương thức được thực hiện theo kiểu an toàn luồng. Kỹ thuật đảm bảo rằng phương thức service không được gọi đồng thời cho những trường hợp không thể thực hiện.

Khi Servlet không được tải (ví dụ: phiên bản mới được nạp hoặc server đang tắt) thì phương thức destroy() được gọi. Khi phương thức destroy được gọi, thì các luồng thực hiện phương thức



Hình 3.5 Vòng đời Servlet đặc trưng

service vẫn còn đang hoạt động, chính vì thế `destroy` cần phải có sự an toàn luồng. Tất cả các tài nguyên đó được phân bổ trong `init` nên sẽ được giải phóng trong `destroy`. Phương thức này được đảm bảo chỉ gọi duy nhất một lần trong một vòng đời Servlet.

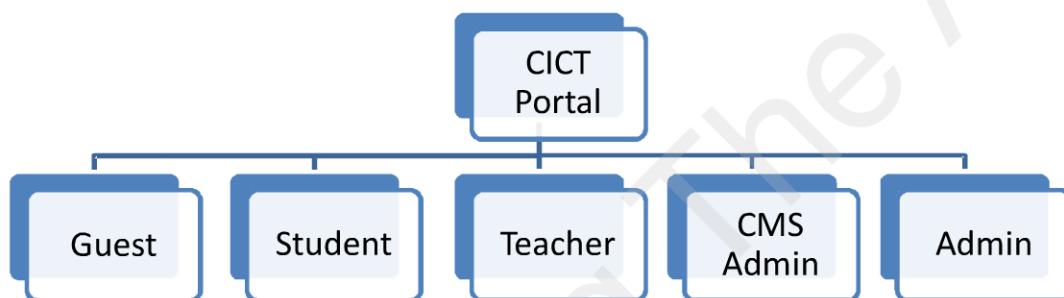
Chương 3: Ứng dụng thực tiễn

3.1 Xây dựng cổng thông tin CICT:

CICT Portal được xây dựng dựa trên cơ sở về cấu trúc và nội dung của Khoa Công Nghệ Thông Tin – Truyền Thông của Trường Đại Học Cần Thơ.

3.1.1 Đối tượng người dùng trong hệ thống CICT Portal

Hệ thống có các dạng người dùng sau:



Hình 4. 1 Sơ đồ khái quát về hệ thống người dùng trong PQID Portal

- Người dùng khách (Guest): là người dùng truy xuất portal mà không đăng ký, chỉ truy xuất đến các tài nguyên công cộng của người dùng đã đăng ký, của cộng đồng.
- Sinh viên (Student): là người dùng đã đăng ký tài khoản trên portal, có thể truy xuất đến các tài nguyên công cộng như người dùng khách, tham gia vào các cộng đồng, có thể gửi luồng mới trong các chuyên mục, trả lời trong diễn đàn của cộng đồng.
- Giảng Viên (Teacher): người dùng đã đăng ký tài khoản trên portal, cũng có quyền truy xuất đến các nơi mà sinh viên truy xuất trên portal. Ngoài ra, giảng viên còn có tài nguyên riêng, có thể gửi bài viết lên Portal, nhưng phải được thông qua quản trị viên hệ quản trị nội dung.
- Quản trị viên hệ quản trị nội dung (CMS Admin): cập nhật thông tin cho portal, được phép thêm, sửa, xóa nội dung của các trang mà người dùng khách có thể truy xuất. Quản lý thư viện hình ảnh, tài liệu của hệ thống.Thêm vào đó, CMS Admin còn có quyền kiểm duyệt các bài viết do giảng viên gửi để đăng trên trang portal.
- Quản trị hệ thống (Admin): là người có toàn quyền trong hệ thống, có thể thực hiện tất cả các chức năng, tất cả các hành động thuộc portal. Trách

nhiệm quan trọng của Admin đó chính là cấp quyền cho các người dùng cho phù hợp.

3.1.2 Cài đặt các trang trong hệ thống



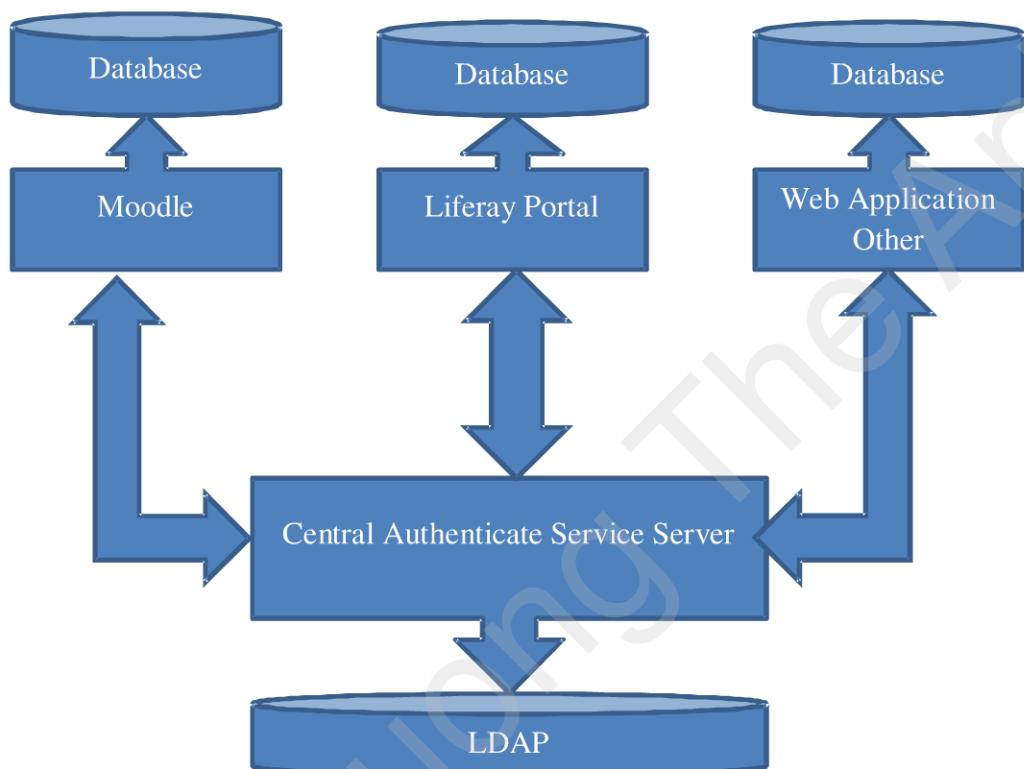
Hình 4. 2 Cấu Trúc Trang

Cấu trúc trang trong hệ thống CICT Portal:

- **Thông tin chung:** Giới thiệu tổng quan, quá trình phát triển, sở đồ tổ chức, Đảng Ủy Khoa, cùng với ban chủ nhiệm khoa Công Nghệ Thông Tin và Truyền Thông Đại Học Cần Thơ
- **Nghiên Cứu Khoa Học và Ăn Phẩm:** cập nhật thông tin những công trình nghiên cứu khoa học từ năm 1985 đến nay, thông tin về các hội thảo về công nghệ thông tin trong và ngoài nước, bên cạnh đó cũng thông tin về các cuộc tổ chức chuyên đề nghiên cứu.
- **Hợp Tác:** thông tin về mối quan hệ giữa khoa với các tổ chức, doanh nghiệp, trường đại học trong và ngoài nước.
- **Các Đơn Vị Trực Thuộc:** giới thiệu về các bộ môn của khoa, cùng với các trung tâm, tổ kỹ thuật và văn phòng khoa.
- **Chương Trình Đào Tạo:** cho biết chi tiết về các môn học được đào tạo tương ứng với từng hệ đào tạo như Đại Học, Cao Đẳng, Thạc Sĩ, Vừa Học Vừa Làm,...
- **Thông Tin Đào Tạo:** thông tin về việc tuyển sinh đào tạo Thạc Sĩ, Tiến Sĩ tại khoa, thông báo dành cho việc đào tạo sau đại học, đặc biệt giới thiệu về việc đào tạo song ngữ của Khoa.
- **Đào Tạo Ngắn Hạn:** thông tin về các khóa đào tạo ngắn hạn như đào tạo tổ chức thi cấp chứng chỉ tin học ứng dụng A – B, mở các lớp Kỹ Thuật Phần Cứng, Quản Trị Mạng.
- **Tin Tức:** các tin tức, thông báo cho sinh viên từ khoa, hay các giảng viên, thông tin về học bổng cho sinh viên, thông tin việc làm cho sinh viên sắp ra trường.
- **Góc Học Thuật:** là một góc dành cho các sinh viên chia sẻ kinh nghiệm, học tập lẫn nhau. Khoa cũng có thành lập một câu lạc bộ Tin Học, để giúp các bạn rèn luyện kỹ năng thực hành, thì đây là nơi thông tin những thông báo chuyên đề mà câu lạc bộ tổ chức.
- **Download:** cung cấp những tập tin về các biểu mẫu, đơn từ, quyết định để người dùng có thể tải về khi cần.
- **Góc Sinh Viên:** thông tin về các hoạt động thể thao văn nghệ dành cho sinh viên, thông tin về các đề tài niêm luận, tiểu luận, luận văn của các khóa trước cho các sinh viên tham khảo.

3.2 Triển khai Central Authentication Service (CAS):

Phần này sẽ hướng dẫn việc triển khai CAS cho các ứng dụng: cấu hình CAS Server, cấu hình ứng dụng để giao tiếp với CAS Server.



Hình 4.3 Mô hình triển khai CAS Server – Liferay - Moodle

Ở mô hình trên, triển khai CAS với Tomcat Server. Để Tomcat hỗ trợ SSL cho CAS, do CAS sử dụng giao thức SSL, thì cần thực hiện một số hướng dẫn sau:

- Tạo SSL Certificate với keytool JAVA.
 - Tạo keystore



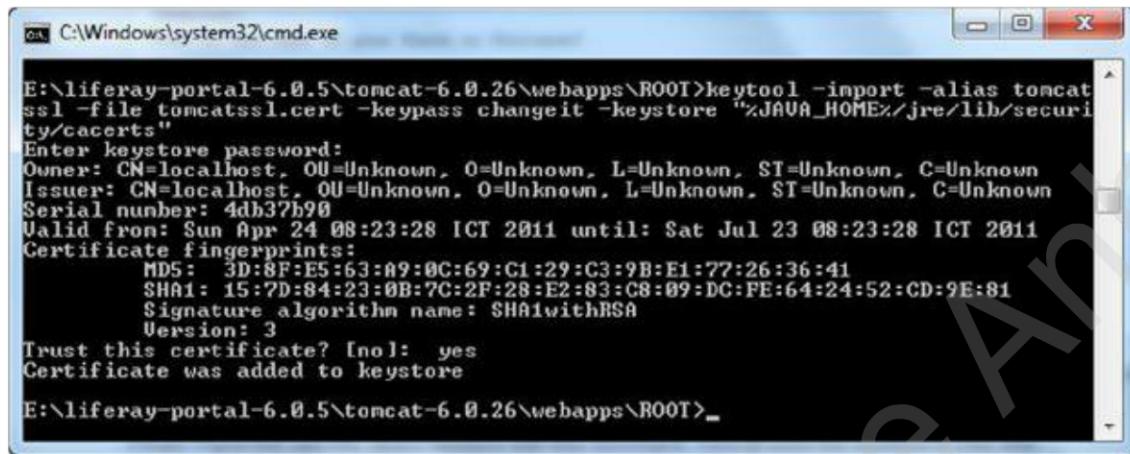
Hình 4.4 Tạo keystore

- Xuất ra file Certificate từ keystore cá nhân (trong windows lưu trữ keystore cá nhân này tại C:\Users\<username>\keystore.)



Hình 4.5 Xuất ra tập tin tomcatssl.cert

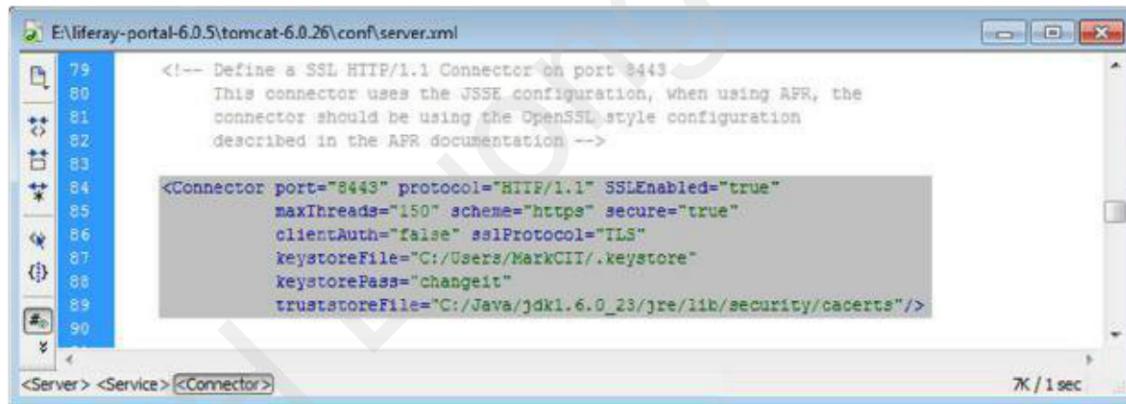
- Cuối cùng, thêm tập tin certificate đó vào keystore của JAVA. Do Tomcat sử dụng keystore trong JRE.



```
C:\Windows\system32\cmd.exe
E:\liferay-portal-6.0.5\tomcat-6.0.26\webapps\ROOT>keytool -import -alias tomcat
ssl -file tomcatssl.cert -keystore "%JAVA_HOME%\jre\lib\security\cacerts"
Enter keystore password:
Owner: CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
Issuer: CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
Serial number: 4db37b90
Valid from: Sun Apr 24 08:23:28 ICT 2011 until: Sat Jul 23 08:23:28 ICT 2011
Certificate fingerprints:
MD5: 3D:8F:E5:63:A9:0C:69:C1:29:C3:9B:E1:77:26:36:41
SHA1: 15:7D:84:23:0B:7C:2F:28:E2:83:C8:09:DC:FE:64:24:52:CD:9E:81
Signature algorithm name: SHA1withRSA
Version: 3
Trust this certificate? [no]: yes
Certificate was added to keystore
E:\liferay-portal-6.0.5\tomcat-6.0.26\webapps\ROOT>
```

Hình 4.6 Import tập tin tomcatssl.cert vào keystore của JAVA

- Vào thư mục conf của tomcat, tìm tập tin server.xml. Bỏ comment tại dòng Connector port="8443" (SSL). Thêm vào những tham số cho keystoreFile, keystorePass, truststoreFile, SSLEnabled=true.



Hình 4.7 Cấu Hình server.xml của Tomcat

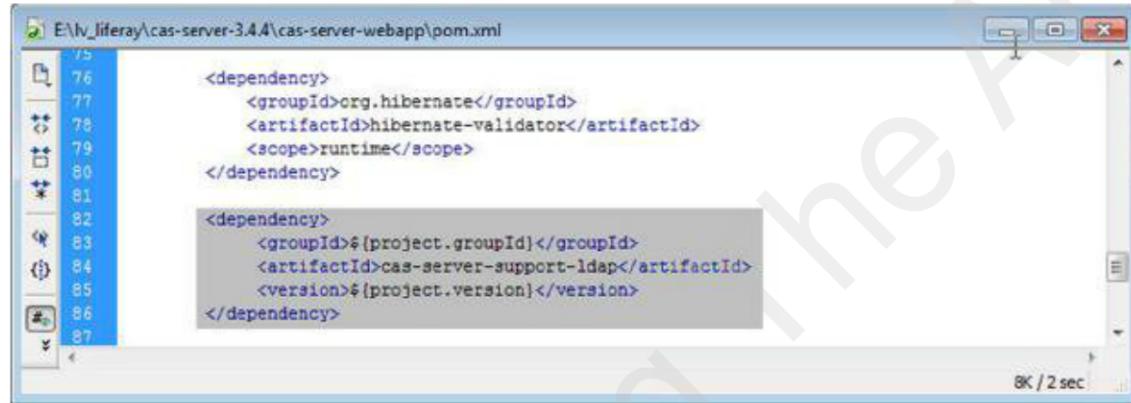
Đến đây, tomcat Server đã có thể hỗ trợ SSL cho CAS.

3.2.1 Triển khai cơ chế xác thực CAS – LDAP:

CAS hỗ trợ nhiều cơ chế xác thực như: LDAP directory, JDBC, RADIUS, Trusted, X.509 Certificate, Active Directory,... Trong phần này, sẽ hướng dẫn và cài đặt cơ chế xác thực CAS – LDAP.

Sau khi tải CAS Server phiên bản 3.4.4 tại <http://downloads.jasig.org/cas/cas-server-3.4.4-release.zip>. Giải nén tập tin vừa tải về ta được thư mục cas-server-3.4.4.

- Mở tập tin pom.xml trong thư mục cas-server-3.4.4/cas-server-webapp, thêm một số dòng sau để CAS Server có thể hỗ trợ cơ chế xác thực qua LDAP.



```
E:\liferay\cas-server-3.4.4\cas-server-webapp\pom.xml
  75
  76      <dependency>
  77          <groupId>org.hibernate</groupId>
  78          <artifactId>hibernate-validator</artifactId>
  79          <scope>runtime</scope>
  80      </dependency>
  81
  82      <dependency>
  83          <groupId>${project.groupId}</groupId>
  84          <artifactId>cas-server-support-ldap</artifactId>
  85          <version>${project.version}</version>
  86      </dependency>
  87
```

Hình 4.8 Tập tin pom.xml

- Ké đên mở tập tin deployerConfigContext.xml trong thư mục cas-server-3.4.4\cas-server-webapp\src\main\webapp\WEB-INF và chỉnh sửa lại một số dòng.

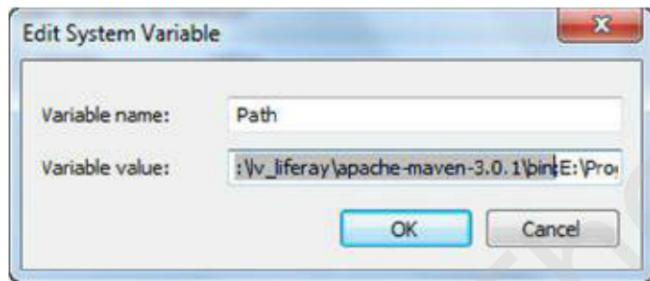


```
88 implements your
89
90 modules.
91
92     +-->
93     <!--
94     <bean
95         class="org.jasig.cas.authentication.handler.SimpleTestUsernamePasswordAuthenticationHandler" />
96
97
98     <bean class="org.jasig.cas.adapters.ldap.BindLdapAuthenticationHandler" >
99         <property name="filter" value="uid={u}" />
100        <property name="searchBase" value="ou=people,dc=dtanh,dc=cit,dc=vn" />
101        <property name="contextSource" ref="contextSource" />
102    </bean>
103
104    </list>
105  </property>
106 </bean>
107
108 <bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
109     <property name="pooled" value="true"/>
110     <property name="url">
111         <list><value>ldap://192.168.1.72:389</value></list>
112     </property>
113     <property name="userDn" value="cn=admin,dc=dtanh,dc=cit,dc=vn"/>
114     <property name="password" value="secret"/>
115 </bean>
116
```

Hình 4.9 Chỉnh sửa tập tin deployerConfigContext.xml

Ghi chú: LDAP Server được cài đặt với địa chỉ 192.168.1.72. Trong phần phụ lục sẽ có hướng dẫn cụ thể về thiết lập một LDAP Server như thế nào.

- Build lại CAS Server với Maven. Cài đặt Maven tương đối đơn giản, tải về tập tin có đuôi .ZIP tại <http://apache.cs.utah.edu/maven/binaries/apache-maven-3.0.3-bin.zip>, việc tiếp theo là giải nén và thiết lập biến môi trường cho maven.



Hình 4. 10 Thiết lập biến môi trường path cho maven

Khi đã cài đặt maven, di chuyển đến thư mục cas-server-3.4.4\cas-server-webapp và sử dụng lệnh mvn package install để build lại.

```
C:\Windows\system32\cmd.exe
1-plugin/2.3/maven-install-plugin-2.3.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/plugins/maven-install-plugin/2.3/maven-install-plugin-2.3.pom (5 KB at 4.0 KB/sec)
Downloading: http://repo1.maven.org/maven2/org/apache/maven/plugins/maven-plugins/13/maven-plugins-13.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/plugins/maven-plugins/13/maven-plugins-13.pom (12 KB at 11.5 KB/sec)
Downloading: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-utils/1.5.6/plexus-utils-1.5.6.jar
Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-utils/1.5.6/plexus-utils-1.5.6.jar (245 KB at 16.9 KB/sec)
[INFO] Installing E:\\v_liferay\\cas-server-3.4.4\\cas-server-webapp\\target\\cas.war to C:\\Users\\MarkCIT\\.m2\\repository\\org\\jasig\\cas\\cas-server-webapp\\3.4.4\\cas-server-webapp-3.4.4.war
[INFO] Installing E:\\v_liferay\\cas-server-3.4.4\\cas-server-webapp\\pom.xml to C:\\Users\\MarkCIT\\.m2\\repository\\org\\jasig\\cas\\cas-server-webapp\\3.4.4\\cas-server-webapp-3.4.4.pom
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 20:40.476s
[INFO] Finished at: Sun Apr 24 12:15:52 ICT 2011
[INFO] Final Memory: 6M/15M
[INFO]
E:\\v_liferay\\cas-server-3.4.4\\cas-server-webapp>
```

Hình 4. 11 Build CAS Server qua Maven đã thành công

- Sau khi build thành công chép tập tin cas-web.war trong thư mục cas-server-3.4.4\cas-server-webapp\target vào thư mục tomcat-6.0.26\webapps của Tomcat Server.

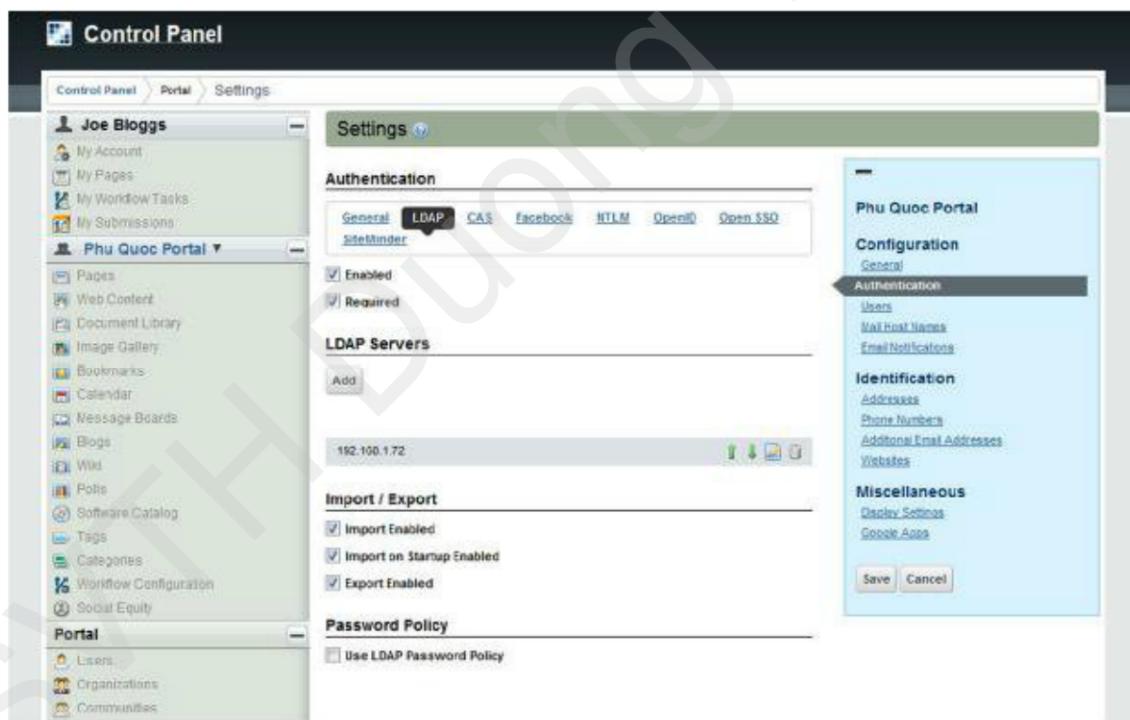
Name	Date modified	Type	Size
cas-server-webapp-3.4.4	2/11/2011 3:39 PM	File folder	
classes	2/11/2011 3:39 PM	File folder	
maven-archiver	2/11/2011 3:39 PM	File folder	
surefire-reports	2/11/2011 3:39 PM	File folder	
test-classes	2/11/2011 3:39 PM	File folder	
cas-web.war	2/11/2011 3:40 PM	WAR File	19,413 KB

Hình 4.12 Thư mục target

- Cuối cùng là chạy Tomcat để deploy cas-web.war.

3.2.2 Cấu hình Liferay sử dụng CAS và LDAP:

Để cấu hình CAS với Liferay và LDAP với Liferay thì cần phải đăng nhập với quyền Admin. Vào Control Panel → Settings → Authentication.



Hình 4.13 Giao diện cấu hình Authentication trong liferay

3.2.2.1 Liferay và LDAP

Chọn tab LDAP để tiến hành cấu hình Liferay với LDAP.

- Authentication: đánh dấu đồng ý với việc xác thực qua LDAP.

Authentication



Hình 4. 14 Cấu hình Liferay - LDAP

- LDAP Servers: thiết lập những tham số để kết nối đến LDAP Server.
- Đặt tên cho máy chủ LDAP.

Server Name

192.168.1.72

Hình 4. 15 Tên Server

- Kết nối đến LDAP Server.

Connection

Base Provider URL

ldap://192.168.1.72:389

Base DN

dc=dtanh,dc=cit,dc=vn

Principal

cn=admin,dc=dtanh,dc=cit,dc=vn

Credentials

.....

Test LDAP Connection

Hình 4. 16 Thiết lập các thông số connection đến LDAP Server

- Cấu hình các tham số về User.

Users

Authentication Search Filter

(mail=@email_address@)

Import Search Filter

(objectClass=inetOrgPerson)

User Mapping

Screen Name

cn

Password

userPassword

Email Address

mail

Full Name

First Name

givenName

Middle Name

Last Name

sn

Job Title

title

Group

gidNumber

Test LDAP Users

Hình 4. 17 Thiết lập các thông số về LDAP User

- o Các tham số về Group.

Groups

Import Search Filter

(objectClass=posixGroup)

Group Mapping

Group Name

cn

Description

description

User

memberUid

Test LDAP Groups

Hình 4.18 Thiết lập các tham số về Group trong LDAP

- Export từ Liferay qua LDAP.

Export

Users DN

ou=people,dc=dtanh,dc=cit,dc=vn

User Default Object Classes

inetOrgPerson,top

Groups DN

ou=groups,dc=dtanh,dc=cit,dc=vn

Group Default Object Classes

posixGroup,top

Hình 4.19 Một vài thuộc tính đồng bộ từ Liferay đến LDAP

- Import/Export: đồng bộ dữ liệu (tài khoản người dùng) giữa Liferay và LDAP.

Import / Export

- Import Enabled
- Import on Startup Enabled
- Export Enabled

Hình 4. 20 Đồng bộ mỗi khi khởi động Liferay

3.2.2.2 Liferay và CAS:

Tương tự, để cấu hình CAS ta chọn tab CAS để thiết lập cho Liferay giao tiếp với CAS.

Authentication

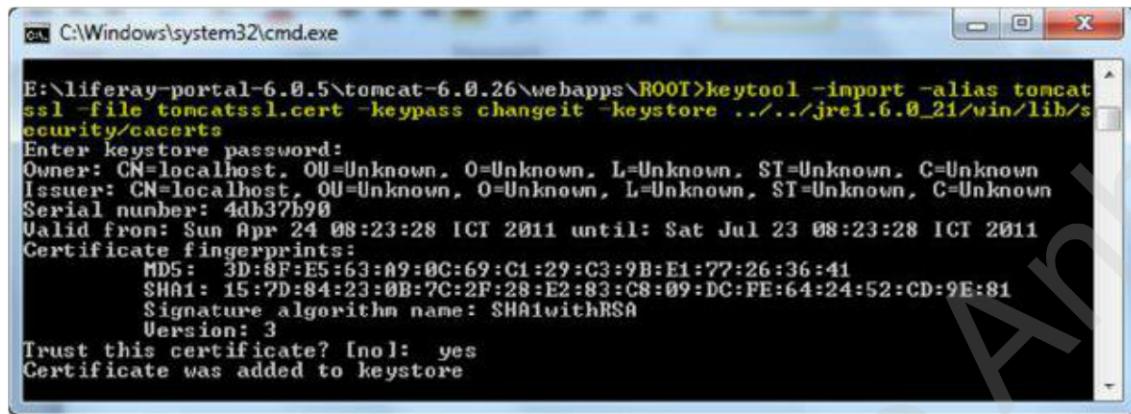
The screenshot shows the 'Authentication' configuration page. At the top, there is a horizontal navigation bar with tabs: General, LDAP, CAS, Facebook, NTLM, OpenID, and Open SSO. The 'CAS' tab is highlighted with a black background and white text. Below the tabs, there are several configuration fields:

- Enabled:** A checkbox with the label 'Enabled' is checked.
- Import from LDAP:** A checkbox with the label 'Import from LDAP' is checked.
- Login URL:** A text input field containing 'https://localhost:8443/cas-web/login'.
- Logout URL:** A text input field containing 'https://localhost:8443/cas-web/logout?url=http://localhost:8080'.
- Server Name:** A text input field containing 'localhost:8080'.
- Server URL:** A text input field containing 'https://localhost:8443/cas-web'.
- Service URL:** A text input field containing 'http://localhost:8080/c/portal/login'.

Hình 4. 21 Thiết lập CAS với Liferay

Ghi chú: ở phần Logout URL: <https://localhost:8443/cas-web/logout?url=http://localhost:8080>. Nhằm mục đích, khi đăng xuất với tham số url như thế thì sau khi hủy ticket xong sẽ chuyển sang trang người dùng khách của liferay.

Ngoài ra cũng cần phải thêm tập tin certificate tạo ban đầu vào trong keystore của JRE Liferay như hình bên dưới.

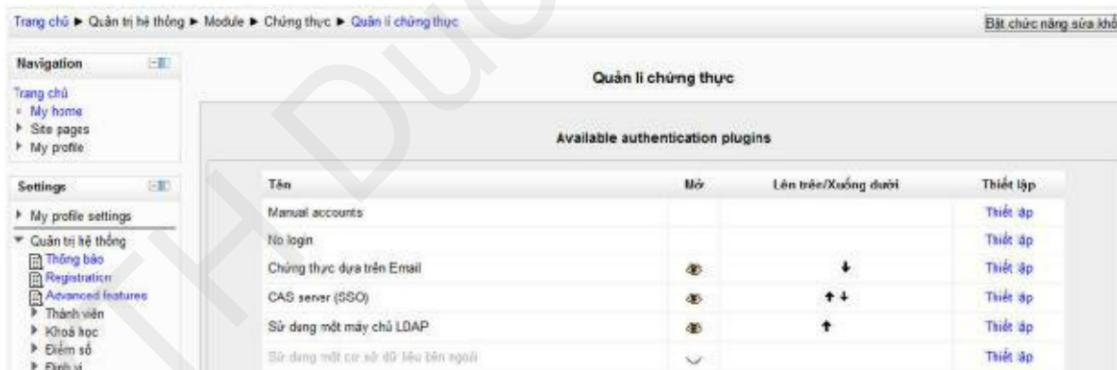


```
C:\Windows\system32\cmd.exe
E:\liferay-portal-6.0.5\tomcat-6.0.26\webapps\ROOT>keytool -import -alias tomcat
ssl -file tomcatssl.cert -keystore ../../jre1.6.0_21/win/lib/security/cacerts
Enter keystore password:
Owner: CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
Issuer: CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
Serial number: 4db37b90
Valid from: Sun Apr 24 08:23:28 ICT 2011 until: Sat Jul 23 08:23:28 ICT 2011
Certificate fingerprints:
MD5: 3D:8F:E5:63:A9:0C:69:C1:29:C3:9B:E1:77:26:36:41
SHA1: 15:2D:84:23:0B:7C:2F:28:E2:83:C8:09:DC:FE:64:24:52:CD:9E:81
Signature algorithm name: SHA1withRSA
Version: 3
Trust this certificate? [no]: yes
Certificate was added to keystore
```

Hình 4. 22 Import certificate vào JRE của Liferay

3.2.3 Triển khai CAS với Moodle:

Trước tiên cần phải tải Moodle về tại địa chỉ <http://download.moodle.org/download.php/stable20/moodle-latest-20.zip> và tiến hành cài đặt để chạy được. Chi tiết cài đặt có thể tham khảo trên website http://docs.moodle.org/en/Installing_Moodle. Sau khi đã cài đặt xong, thì đăng nhập với quyền Admin vừa tạo. Vào Site Administration → Plugins → Authentication → Manage authentication.



Hình 4. 23 Giao diện Manage Authentication

Sau khi đã bật cơ chế xác thực qua CAS Server như hình bên trên. Thì bước tiếp theo là thiết lập các thông số để kết nối đến với CAS Server:

- Cấu hình địa chỉ và cổng giao tiếp với CAS Server.

CAS server configuration

Hostname:	localhost	Hostname of the CAS server eg: host.domain.fr
Base URI:	cas-web/	URI of the server (nothing if no baseURL) For example, if the CAS server responds to host.domain.fr/CAS/ then cas_baseuri = CAS/
Port:	8443	Port of the CAS server
Version:	CAS 2.0	Version of CAS
Language:	english	Selected language
Proxy mode:	No	Turn this to 'yes' if you use CAS in proxy-mode
Logout CAS:	Yes	Turn this to 'yes' if you want to logout from CAS when you disconnect from Moodle
Multi-authentication:	No	Turn this to 'yes' if you want to have multi-authentication (CAS + other authentication)
Server validation:	No	Turn this to 'yes' if you want to validate the server certificate
Certificate path:		Path of the CA chain file (PEM Format) to validate the server certificate

Hình 4.24 Một số tham số về cấu hình CAS Server

➤ Thiết lập LDAP Server

LDAP server settings

Host URL:	ldap://192.168.1.72:389	Specify LDAP host in URL-form like 'ldap://ldap.myorg.com/' or 'ldaps://ldap.myorg.com/'. Separate multiple servers with ';' to get failover support.
Version:	3	The version of the LDAP protocol your server is using.
LDAP encoding:	utf-8	Specify encoding used by LDAP server. Most probably utf-8, MS AD v2 uses default platform encoding such as cp1252, cp1250, etc.

Hình 4.25 Thiết lập LDAP Server

➤ Thiết lập kết nối

Bind settings

Distinguished name:	cn=admin,dc=dtanh,dc=cit,dc=m	If you want to use bind-user to search users, specify it here. Something like 'cn=ldapuser,ou=public,o=org'
Password:	*****	Password for bind-user.

Hình 4.26 Kết nối đến LDAP

➤ Thiết lập việc tìm kiếm người dùng trong LDAP

User lookup settings

User type: posixAccount (rfc2307)

Select how users are stored in LDAP. This setting also specifies how login expiration, grace logins and user creation will work.

Contexts: ou=people,dc=dtanh,dc=cit,dc=vn

List of contexts where users are located. Separate different contexts with ;. For example: 'ou=users,o=org; ou=others,o=org'

Search subcontexts: Yes

Search users from subcontexts.

Dereference aliases: No

Determines how aliases are handled during search. Select one of the following values: "No" (LDAP_DEREF_NEVER) or "Yes" (LDAP_DEREF_ALWAYS)

User attribute: cn

Optional: Overrides the attribute used to name/search users. Usually 'cn'.

Member attribute: member

Optional: Overrides user member attribute, when users belongs to a group. Usually 'member'

Member attribute uses: 1

Optional: Overrides handling of member attribute values, either 0 or 1

Object class: inetOrgPerson

Optional: Overrides objectClass used to name/search users on ldap_user_type. Usually you don't need to change this.

Course creator

Attribute creators:

Group creators: cn=teacher ou=groups,dc=dtanh,dc=cit,dc=vn

Cron synchronization script

Removed ext user: Keep internal

Specify what to do with internal user account during mass synchronization when user was removed from external source. Only suspended users are automatically revived if they reappear in ext source.

Hình 4.27 Thiết lập về người dùng LDAP

- Data Mapping: tương tự như các tham số là các kiểu dữ liệu tương đồng với nhau.

Data mapping

First name	givenName	These fields are optional. You can choose to pre-fill some Moodle user fields with information from the LDAP fields that you specify here. If you leave these fields blank, then nothing will be transferred from LDAP and Moodle defaults will be used instead.
	Update local: On creation	In either case, the user will be able to edit all of these fields after they log in.
	Update external: Never	
	Lock value: Unlocked	
Surname	sn	Note: Updating external LDAP data requires that you set binddn and bindpw to a bind-user with editing privileges to all the user records. It currently does not preserve multi-valued attributes, and will remove extra values on update.
	Update local: On creation	
	Update external: Never	
	Lock value: Unlocked	
Email address	mail	Note: Updating external LDAP data requires that you set binddn and bindpw to a bind-user with editing privileges to all the user records. It currently does not preserve multi-valued attributes, and will remove extra values on update.
	Update local: On creation	
	Update external: Never	
	Lock value: Unlocked	
City/town	city	
	Update local: On creation	
	Update external: Never	
	Lock value: Unlocked	
Country	country	
	Update local: On creation	
	Update external: Never	
	Lock value: Unlocked	
Language	language	
	Update local: On creation	
	Update external: Never	
	Lock value: Unlocked	

Hình 4.28 Data Mapping

Việc cuối cùng là lưu lại nhưng thiết lập đó và chạy thử.

3.2.4 Triển khai CAS với Servlet:

Các nhà phát triển Servlet không cần bận tâm nhiều về cơ chế chứng, chỉ việc sử dụng cơ chế chứng thực sẵn có của hệ thống nào sử dụng CAS. Bằng cách rất đơn giản và nhẹ nhàng là thêm một số dòng như hình bên dưới vào tập tin web.xml của mã nguồn Servlet cần đưa vào hệ thống.

```
<filter>
<filter-name>CAS Filter</filter-name>
<filter-class>edu.yale.its.tp.cas.client.filter.CASFilter</filter-class>
<init-param>
    <param-name>edu.yale.its.tp.cas.client.filter.loginUrl</param-name>
    <param-value>https://localhost:8443/cas-web/login</param-value>
</init-param>
<init-param>
    <param-name>edu.yale.its.tp.cas.client.filter.validateUrl</param-name>
    <param-value>https://localhost:8443/cas-web/proxyValidate</param-value>
</init-param>
<init-param>
    <param-name>edu.yale.its.tp.cas.client.filter.serverName</param-name>
    <param-value>localhost:8084</param-value>
</init-param>
</filter>
```

```
<filter>
<filter-name>Servlet Mapped Filter</filter-name>
<filter-class>filters.ExampleFilter</filter-class>
<init-param>
    <param-name>attribute</param-name>
    <param-value>filters.ExampleFilter.SERVLET_MAPPED</param-value>
</init-param>
</filter>
<filter>
    <filter-name>Path Mapped Filter</filter-name>
    <filter-class>filters.ExampleFilter</filter-class>
```

```
<filter-mapping>
    <filter-name>CAS Filter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>

<filter-mapping>
    <filter-name>Servlet Mapped Filter</filter-name>
    <servlet-name>invoker</servlet-name>
</filter-mapping>
<filter-mapping>
    <filter-name>Path Mapped Filter</filter-name>
    <url-pattern>/servlet/*</url-pattern>
</filter-mapping>

<jsp-config>
    <jsp-property-group>
        <url-pattern>*.jsp</url-pattern>
        <page-encoding>UTF-8</page-encoding>
    </jsp-property-group>
</jsp-config>
```

Hình 4. 29 Thiết lập CAS và Servlet

Chương 4: Kết luận

4.1 Kết quả đạt được

- ✓ Tìm hiểu và xây dựng một portal dựa trên Liferay Portal.
- ✓ Nghiên cứu và triển khai thành công kỹ thuật Single Sign On là CAS.
- ✓ Single Sign On thành công cho Liferay, Moodle, Servlet.
- ✓ Ứng dụng moodle vào hệ thống chứng thực một cửa trên Liferay Portal.
- ✓ Nghiên cứu và thiết lập được LDAP Server trên Ubuntu.

4.2 Hạn chế

- Giao diện web vẫn chưa được chú trọng nhiều.
- Cài đặt Single Sign On chưa hoàn toàn đầy đủ tính năng.
- Còn nhiều vấn đề chưa hiểu rõ về Liferay, cũng như Moodle.
- Chưa đồng bộ được người dùng trong LDAP và Moodle.
- Còn nhiều mặt hạn chế trong quá trình thực hiện, báo cáo tiến độ công việc.

4.3 Hướng phát triển:

- Phát triển cổng thông tin khoa Công Nghệ Thông Tin Và Truyền Thông – Trường Đại Học Cần Thơ vào thực tiễn.
- Phát triển các portlet riêng cho CICT Portal.
- Triển khai một chính sách phân quyền cho người dùng chặt chẽ hơn.

Phụ lục

Tài Liệu Tham Khảo

- [1] **Liferay Portal 6 Enterprise Intranets - Jonas X. Yuan**
- [2] **Practical Liferay Java™-based Portal Applications Development - Poornachandra Sarang, Ph.D.**
- [3] **Liferay Docs and Resources**
<http://www.liferay.com/documentation/liferay-portal/6.0/getting-started>
- [4] **Liferay Community Wikis** - <http://www.liferay.com/community/wiki>
- [5] **Central Authentication Service (CAS)**
<http://www.jusfortechies.com/cas/miscellaneous.html>
- [6] **CAS Server (JASIG Community)** - <http://www.jasig.org/cas/download>
- [7] **Moodle** - <http://docs.moodle.org/>
- [8] **Java Servlet** - <http://mobilesprogramming.wordpress.com/2010/08/18/tim-hi%BB%83u-servlet-ph%E1%BA%A7n-1/>
- [9] **LDAP Wikipedia** - <http://en.wikipedia.org/wiki/LDAP>
- [10] **Ubuntu 10.04 LTS Configurations** - http://server-world.info/en/note?os=Ubuntu_10.04&p=install.
- [11] **Understanding and Deploying LDAP Directory Services, Second Edition** - *Timothy A. Howes Ph.D., Mark C. Smith, Gordon S. Good.*
- [12] **CAS 2.0 Protocol Specification v1.0.doc** - *Drew Mazurek drew.mazurek@yale.edu* download tại <https://source.jasig.org/cas3/trunk/etc/docs/CAS+2.0+Protocol+Specification+v1.0.doc>

Hướng dẫn cài đặt LDAP Server trên Ubuntu 10.10

Để thực hiện được bước này, người thực hiện cần cài đặt thành công ubuntu server (với thời gian viết quyền báo cáo này thì phiên bản sử dụng là 10.04). Sau đây là trình tự tiến hành thiết lập một LDAP Server trên Ubuntu 10.04.

A. Đây là một server, nên cần phải thiết lập địa chỉ ip tĩnh.

```
modadmin@LDAPServer:~$ sudo vi /etc/network/interfaces
```

```
[sudo] password for modadmin:

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
# iface eth0 inet dhcp
#add these lines
iface eth0 inet static
address 192.168.1.72          //định nghĩa địa chỉ IP
network 192.168.1.0            //định nghĩa địa chỉ network
netmask 255.255.255.0          //định nghĩa subnet mask
broadcast 192.168.1.255        //định nghĩa địa chỉ broadcast
gateway 192.168.1.2           //định nghĩa địa chỉ gateway
```

- ✓ Tiến hành khởi động lại giao diện mạng và kiểm tra địa chỉ đã thiết lập:

```
modadmin@LDAPServer:~$ sudo /etc/init.d/networking restart
modadmin@LDAPServer:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:0c:29:28:42:a8
          inet addr:192.168.1.72  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe28:42a8/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:335 errors:0 dropped:0 overruns:0 frame:0
            TX packets:187 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:33566 (33.5 KB)  TX bytes:27824 (27.8 KB)
            Interrupt:19 Base address:0x2000

lo       Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:30 errors:0 dropped:0 overruns:0 frame:0
            TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:2664 (2.6 KB)  TX bytes:2664 (2.6 KB)
```

- B. Việc thứ 2 cần làm là xây dựng một DNS server, nó có nhiệm vụ phân giải tên miền (domain name) hay địa chỉ ip. Điều này rất cần thiết để cho cấu hình cho bộ định tuyến khi các gói TCP hay UDP đến cổng 53 có thể qua được. Trước tiên cần cài đặt Bind.

```
modadmin@LDAPServer:~$ sudo apt-get install bind9
```

Giả sử hệ thống có 3 máy tính cùng DNS server (192.168.1.72) với các địa chỉ và tên lần lượt như sau:

Máy 1

LDAPServer

192.168.1.72

Máy 2	client	192.168.1.132
Máy 3	DHCP	192.168.1.254

Domain name: dtanh.cit.vn. Bổ sung 03 Host (A) record tương ứng với 03 máy tính vào DNS server:

LDAPServer.dtanh.cit.vn	192.168.1.72
client.dtanh.cit.vn	192.168.1.132
DHCP.dtanh.cit.vn	192.168.1.254

✓ Hiệu chỉnh tập tin cấu hình chính của BIND:

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

✓ Hiệu chỉnh tập tin named.conf.options để chuyển (forward) những yêu cầu mà DNS server này không phân giải được.

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/named.conf.options
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    forwarders {
        8.8.8.8;
        8.8.4.4;
    };

    auth-nxdomain no;      # conform to RFC1035
    listen-on-v6 { any; };
};
```

- ✓ Hiệu chỉnh tập tin named.conf.local:

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
include "/etc/bind/zones.rfc1918";
zone "dtanh.cit.vn" {
    type master;
    file "/etc/bind/192.db";
};
```

- ✓ Tạo tập tin /etc/bind/192.db để lưu trữ từng cặp địa chỉ ip và tên máy để DNS server này phân giải.

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/192.db
$TTL 86400
```

```
@ IN SOA LDAPServer.dtanh.cit.vn. root.dtanh.cit.vn. (
    2 ; Serial
    604800 ; Refresh
    86400 ; Retry
    2419200 ; Expire
    86400 ) ; Negative Cache TTL
;
@ IN NS LDAPServer.dtanh.cit.vn.
@ IN A 192.168.1.72

LDAPServer IN A 192.168.1.72
client IN A 192.168.1.132
DHCP IN A 192.168.1.254
```

- ✓ Tạo tập tin /etc/bind/245.db để dùng mục đích phân giải ngược.

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/245.db
$TTL 86400
@ IN SOA LDAPServer.dtanh.cit.vn. root.dtanh.cit.vn. (
    2 ; Serial
    604800 ; Refresh
    86400 ; Retry
    2419200 ; Expire
    86400 ) ; Negative Cache TTL
;
@ IN NS LDAPServer.dtanh.cit.vn.
;
72 IN PTR LDAPServer.dtanh.cit.vn.
132 IN PTR client.dtanh.cit.vn.
254 IN PTR DHCP.dtanh.cit.vn.
```

- ✓ Hiệu chỉnh tập tin zones.rfc1918

```
modadmin@LDAPServer:~$ sudo vi /etc/bind/zones.rfc1918
```

```

zone "10.in-addr.arpa"      { type master; file "/etc/bind/db.empty"; };

zone "16.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "17.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "18.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "19.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "20.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "21.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "22.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "23.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "24.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "25.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "26.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "27.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "28.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "29.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "30.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "31.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };

#zone "168.192.in-addr.arpa" { type master; file "/etc/bind/db.empty"; };

zone "1.168.192.in-addr.arpa" { type master; file "/etc/bind/245.db"; };

```

- ✓ Hiệu chỉnh tập tin /etc/resolv.conf để liệt kê danh sách các DNS server trong mạng của mình:

```

modadmin@LDAPServer:~$ sudo vi /etc/resolv.conf
nameserver 192.168.1.72
domain dtanh.cit.vn
search dtanh.cit.vn

```

- ✓ Khởi động lại dịch vụ BIND:

```

modadmin@LDAPServer:~$ sudo /etc/init.d/bind9 restart
[sudo] password for modadmin:
* Stopping domain name service... bind9                                [ OK ]
* Starting domain name service... bind9                                [ OK ]

```

- ✓ Bước cuối cùng là thực hiện kiểm tra xem DNS hoạt động đúng đắn không:

```

modadmin@LDAPServer:~$ dig DHCP.dtanh.cit.vn

; <>> DiG 9.7.0-P1 <>> DHCP.dtanh.cit.vn
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47113
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;DHCP.dtanh.cit.vn.           IN      A

;; ANSWER SECTION:
DHCP.dtanh.cit.vn.       86400   IN      A      192.168.1.254

;; AUTHORITY SECTION:
dtanh.cit.vn.            86400   IN      NS     LDAPServer.dtanh.cit.vn.


```

```
;; ADDITIONAL SECTION:  
LDAPServer.dtanh.cit.vn. 86400 IN A 192.168.1.72  
  
;; Query time: 1 msec  
;; SERVER: 192.168.1.72#53(192.168.1.72)  
;; WHEN: Fri Dec 3 14:39:21 2010  
;; MSG SIZE rcvd: 92  
  
modadmin@LDAPServer:~$ dig -x 192.168.1.254  
  
; <>> DIG 9.7.0-P1 <>> -x 192.168.1.254  
;; global options: +cmd  
;; Got answer:  
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22416  
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1  
  
;; QUESTION SECTION:  
;254.1.168.192.in-addr.arpa. IN PTR  
  
;; ANSWER SECTION:  
254.1.168.192.in-addr.arpa. 86400 IN PTR DHCP.dtanh.cit.vn.  
  
;; AUTHORITY SECTION:  
1.168.192.in-addr.arpa. 86400 IN NS LDAPServer.dtanh.cit.vn.  
  
;; ADDITIONAL SECTION:  
LDAPServer.dtanh.cit.vn. 86400 IN A 192.168.1.72  
  
;; Query time: 0 msec  
;; SERVER: 192.168.1.72#53(192.168.1.72)  
;; WHEN: Fri Dec 3 14:39:43 2010  
;; MSG SIZE rcvd: 116
```

C. Đến đây, chúng ta sẽ bắt đầu xây dựng một LDAP server để chia sẻ những tài khoản của người dùng cho những mạng cục bộ. Trước tiên cài đặt gói openldap cho server:

```
modadmin@LDAPServer:~$ sudo apt-get install slapd ldap-utils
```

- ✓ Tiếp theo là thêm một vài tập tin schema cần được nạp vào:

```
modadmin@LDAPServer:~$ sudo ldapadd -Y EXTERNAL -H ldapi:/// -f  
/etc/ldap/schema/cosine.ldif  
SASL/EXTERNAL authentication started  
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth  
SASL SSF: 0  
adding new entry "cn=cosine,cn=schema,cn=config"  
  
modadmin@LDAPServer:~$ sudo ldapadd -Y EXTERNAL -H ldapi:/// -f  
/etc/ldap/schema/nis.ldif  
SASL/EXTERNAL authentication started  
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth  
SASL SSF: 0
```

```

adding new entry "cn=nis,cn=schema,cn=config"

modadmin@LDAPServer:~$ sudo ldapadd -Y EXTERNAL -H ldapi:/// -f
/etc/ldap/schema/inetorgperson.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "cn/inetorgperson,cn=schema,cn=config"

```

- ✓ Ké đến là tạo một tập tin LDIF, đặt tên là /etc/ldap/backend.ldif với nội dung như sau:

```

modadmin@LDAPServer:~$ sudo vi /etc/ldap/backend.ldif
# create new
# replace to your domain
# ex: dtanh.cit.vn dc=dtanh,dc=cit,dc=vn
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulepath: /usr/lib/ldap
olcModuleload: back_hdb

dn: olcDatabase=hdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcSuffix: dc=dtanh,dc=cit,dc=vn                                //thay thế domain của bạn
olcDbDirectory: /var/lib/ldap
olcRootDN: cn=admin,dc=dtanh,dc=cit,dc=vn                         //thay thế domain của bạn
olcRootPW: secret                                                 //mật khẩu mà bạn chọn
olcDbConfig: set_cachesize 0 2097152 0
olcDbConfig: set_lk_max_objects 1500
olcDbConfig: set_lk_max_locks 1500
olcDbConfig: set_lk_max_lockers 1500
olcDbIndex: objectClass eq
olcLastMod: TRUE
olcDbCheckpoint: 512 30
olcAccess: to attrs=userPassword by dn="cn=admin,dc=dtanh,dc=cit,dc=vn" write
by anonymous auth by self write by * none                          //thay thế domain của bạn
olcAccess: to attrs=shadowLastChange by self write by * read
olcAccess: to dn.base="" by * read
olcAccess: to * by dn="cn=admin,dc=server,dc=world" write by * read      //thay
thế domain                                         của bạn
                                         của bạn

```

- ✓ Thêm tập tin /etc/ldap/backend.ldif này vào:

```

modadmin@LDAPServer:~$ sudo ldapadd -Y EXTERNAL -H ldapi:/// -f
/etc/ldap/backend.ldif

```

- ✓ Tương tự, tạo thêm các tập tin /etc/ldap/frontend.ldif và /etc/ldap/user.ldif:

```
modadmin@LDAPServer:~$ sudo vi /etc/ldap/frontend.ldif
# create new
# replace to your domain on the line "dn"
# ex: dtanh.cit.vn . dc=dtanh,dc=cit,dc=vn
dn: dc=dtanh,dc=cit,dc=vn                                //thay thế domain của bạn
objectClass: top
objectClass: dcObject
objectclass: organization
o: Dtanh CIT
dc: Dtanh
description: LDAP Server

dn: cn=admin,dc=dtanh,dc=cit,dc=vn                      //thay thế domain của bạn
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword: secret                                      //mật khẩu mà bạn chọn

dn: ou=people,dc=dtanh,dc=cit,dc=vn                      //thay thế domain của bạn
objectClass: organizationalUnit
ou: people

dn: ou=groups,dc=dtanh,dc=cit,dc=vn                      //thay thế domain của bạn
objectClass: organizationalUnit
ou: groups

modadmin@LDAPServer:~$ sudo vi /etc/ldap/user.ldif
# add a user "dtanh"
dn: uid=dtanh,ou=people,dc=dtanh,dc=cit,dc=vn      //thay thế domain của bạn
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: dtanh
sn: LDAPServer
givenName: dtanh
cn: dtanh LDAPServer
displayName: dtanh LDAPServer
uidNumber: 1000
gidNumber: 1000
userPassword: dtanh                                    //mật khẩu mà bạn chọn
gecos: dtanh LDAPServer
loginShell: /bin/bash
homeDirectory: /home/dtanh
shadowExpire: -1
shadowFlag: 0
shadowWarning: 7
shadowMin: 8
shadowMax: 99999
shadowLastChange: 14809
```

- ✓ Bạn bắt đầu thêm 2 tập tin ldif đó vào: