

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ



ĐỒ ÁN TỐT NGHIỆP
**GIẢI PHÁP XÁC THỰC TẬP TRUNG DỰA TRÊN
CÔNG CỤ MÃ NGUỒN MỞ**

Ngành: An toàn thông tin
Mã số: 7.48.02.02

Sinh viên thực hiện:

Phan Hoàng Trung

Lớp: AT13M

Trần Thị Ngọc

Lớp: AT13N

Người hướng dẫn:

PGS.TS. Lương Thế Dũng

Phó Giám Đốc Học Viện Kỹ Thuật Mật Mã

TP. Hồ Chí Minh, 2021

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ



ĐỒ ÁN TỐT NGHIỆP

**GIẢI PHÁP XÁC THỰC TẬP TRUNG DỰA TRÊN
CÔNG CỤ MÃ NGUỒN MỞ**

Ngành: An toàn thông tin
Mã số: 7.48.02.02

Sinh viên thực hiện:

Phan Hoàng Trung

Lớp: AT13M

Trần Thị Ngọc

Lớp: AT13N

Người hướng dẫn:

PGS.TS. Lương Thế Dũng

Phó Giám Đốc Học Viện Kỹ Thuật Mật Mã

TP. Hồ Chí Minh, 2021

MỤC LỤC

Mục lục	i
Danh mục kí hiệu và viết tắt.....	iii
Danh mục hình vẽ.....	iv
Lời cảm ơn	v
Lời nói đầu	vi
Chương 1. TỔNG QUAN VỀ an toàn với các phương pháp xác thực	1
1.1. Giới thiệu về ứng dụng Web	1
1.1.1. <i>Khái niệm.....</i>	<i>1</i>
1.1.2. <i>Một số hướng phát triển ứng dụng Web.....</i>	<i>2</i>
1.2. Tổng quan về xác thực	4
1.2.1. <i>Một số khái niệm trong xác thực</i>	<i>4</i>
1.2.2. <i>Các bước cơ bản trong xác thực</i>	<i>6</i>
1.3. Các yếu tố xác thực	7
1.4. Các phương pháp xác thực phổ biến.....	8
1.4.1. <i>Phương pháp xác thực dựa trên định danh người dùng và mật khẩu.....</i>	<i>8</i>
1.4.2. <i>Phương pháp xác thực sử dụng giao thức PAP.</i>	<i>10</i>
1.4.3. <i>Phương pháp xác thực sử dụng giao thức CHAP.</i>	<i>11</i>
1.4.4. <i>Phương pháp xác thực sử dụng giao thức Kerberos.....</i>	<i>12</i>
1.4.5. <i>Phương pháp xác thực sử dụng Tokens.....</i>	<i>13</i>
1.4.6. <i>Xác thực bằng sinh trắc học.....</i>	<i>15</i>
1.4.7. <i>Phương pháp xác thực đa yếu tố.....</i>	<i>18</i>
1.5. Tổng kết chương 1.....	19
Chương 2. Giải pháp xác thực với OAuth2	20
2.1. Tổng quan về OAuth 2.0.....	20
2.1.1. <i>Đặt vấn đề.....</i>	<i>20</i>
2.1.2. <i>Lịch sử phát triển.....</i>	<i>21</i>
2.1.3. <i>Giới thiệu và định nghĩa OAuth 2.0</i>	<i>22</i>
2.1.4. <i>Các tính năng bảo mật.....</i>	<i>25</i>
2.2. Xác thực Single Sign On (SSO).....	33
2.2.1. <i>Thuật ngữ SSO.....</i>	<i>33</i>
2.2.2. <i>Định nghĩa SSO</i>	<i>35</i>
2.2.3. <i>Cơ chế hoạt động.....</i>	<i>36</i>
2.2.4. <i>Ưu và nhược điểm của SSO.....</i>	<i>40</i>
2.3. Vấn đề xác thực với OAuth2.....	41
2.4. Tổng kết chương 2	44
Chương 3. ÁP DỤNG TRIỂN KHAI MÔ HÌNH XÁC THỰC TẬP TRUNG	45

3.1. Mô hình triển khai	45
3.2. Quy trình triển khai	46
3.2.1. <i>Xây dựng App Server</i>	47
3.2.2. <i>Xây dựng Client Side App</i>	51
3.2.3. <i>Thực hiện cài đặt xác thực</i>	51
3.3. Kết quả thực nghiệm	59
3.4. Kết luận chương 3	61
Kết luận	62
Tài liệu tham khảo.....	63

DANH MỤC KÍ HIỆU VÀ VIẾT TẮT

OTP	One Time Password
SSO	Single Sign On
XSS	Cross-Site Scripting
MAC	Message Authentication Code
TLS	Transport Layer Security
ATM	Automated Teller Machine
POS	Point Of Sale
IS	Information System
PIN	Personal Identification Number
2FA	Two Factor Authentication
PAP	Passwork Authentication Protocol
CHAP	Challenge Handshake Authentication Protocol
ACK	Acknowledgement
NAK	Negative Acknowledge
AS	Authentication Server
TGT	Ticket Granting Ticket
TGS	Ticket Granting Server
KDC	Key Distribution Center
ST	Service Ticket
3FA	Three-Factor Authentication-
TGC	Ticket Granting Cookie
CAS	Central Authentication Service
CMS	Content Management System
IT	Information Technology
HTML	Hypertext Markup Language
IdP	Identity Provider

DANH MỤC HÌNH VẼ

Hình 1.1. Mô hình các bước xác thực	6
Hình 1.2. Form xác thực người dùng và mật khẩu	8
Hình 1.3. Giao thức PAP.....	10
Hình 1.4. Giao thức CHAP	11
Hình 1.5. Giao thức Kerberos	12
Hình 1.6. Thẻ thông minh	13
Hình 1.7. Xác thực sinh trắc học.....	15
Hình 2.1. Giới thiệu về OAuth 2.0.....	22
Hình 2.2. Luồng giao thức trừu tượng	23
Hình 2.3. Luồng mã ủy quyền.....	27
Hình 2.4. Luồng Implicit.....	29
Hình 2.5. Luồng Resource Owner Credentials	30
Hình 2.6. Luồng hoạt động của Client Authentication	31
Hình 2.7. Làm mới mã thông báo truy cập đã hết hạn.....	32
Hình 2.8. Các tác nhân SSO.....	35
Hình 2.9. Hệ thống nhận dạng liên kết.....	38
Hình 2.10. Xác thực SSO	39
Hình 3.1. Mô hình triển khai xác thực	45
Hình 3.2. Mô hình luồng Single Sign-On	46
Hình 3.3. Giao diện của App Server	52
Hình 3.4. Tài khoản quản trị	52
Hình 3.5. Trang quản trị.....	53
Hình 3.6. Nhập thông tin đăng ký	54
Hình 3.7. Client Identifier và Client Secret	55
Hình 3.8. Giao diện Client Side App	59
Hình 3.9. Đăng nhập xác thực.....	59
Hình 3.10. Thực hiện ủy quyền.....	60
Hình 3.11. Xác thực thành công.....	60

LỜI CẢM ƠN

Trong quá trình thực hiện nghiên cứu và phát triển đề án này, chúng em đã nhận được sự chỉ bảo, giúp đỡ tận tình của giảng viên hướng dẫn là thầy PGS.TS.Lương Thế Dũng – PGD Học viện Kỹ thuật Mật mã. Những lời nói động viên, hành động giúp đỡ hỗ trợ từ người thân và bạn bè xung quanh.

Xin cảm ơn tất cả mọi người đã tạo những điều kiện tốt nhất để chúng em có thể hoàn thành đề án tốt nghiệp này!

SINH VIÊN THỰC HIỆN ĐỀ ÁN

Phan Hoàng Trung
Trần Thị Ngọc

LỜI NÓI ĐẦU

Trong thời đại công nghệ đang ngày càng phát triển cả về phần cứng lẫn phần mềm. Nhu cầu sử dụng ngày một tăng, kéo theo đó là xu hướng các dịch vụ cùng nhau chia sẻ dữ liệu người dùng, một người dùng phải quản lý rất nhiều tài khoản, mật khẩu cho các dịch vụ, ứng dụng mà họ tham gia. Khi người sử dụng phải lưu trữ quá nhiều các thông tin bí mật sẽ dẫn đến những vấn đề như không đảm bảo tính an ninh, an toàn và tăng thêm những chi phí khác. Do vậy nhu cầu về giải pháp xác thực tập trung cho các ứng dụng và dịch vụ này là rất cấp thiết.

Đề án tập trung vào nghiên cứu, tìm hiểu các phương pháp xác thực, giải pháp xác thực với OAuth2. Ngoài ra, đề án cũng trình bày giải pháp SSO giúp người sử dụng chỉ cần dùng một tài khoản và mật khẩu SSO duy nhất có thể sử dụng được tất cả các ứng dụng tin cậy. SSO đã được nhiều tổ chức, doanh nghiệp, công ty trên thế giới nghiên cứu và phát triển, tuy nhiên tại Việt Nam thì đây vẫn là lĩnh vực còn khá mới. Cuối cùng, áp dụng và triển khai mô hình xác thực tập trung vào ứng dụng web.

Nội dung đề án được sẽ chia thành 3 chương chính:

Chương 1: Tổng quan về an toàn với các phương pháp xác thực

Trình bày về định nghĩa, thành phần xác thực, các phương pháp xác thực phổ biến.

Chương 2: Giải pháp xác thực với OAuth2

Trình bày về định nghĩa, lịch sử phát triển, cơ chế hoạt động, những ưu và nhược điểm của OAuth2. Giới thiệu về SSO, cách thức hoạt động và các rủi ro bảo mật có thể mắc phải.

Chương 3: Áp dụng triển khai mô hình xác thực tập trung

Tiến hành xây dựng mô hình triển khai, áp dụng cho ứng dụng web. Sẽ bao gồm quá trình cài đặt thư viện, thành phần cần thiết. Triển khai mô hình xác thực tập trung, đánh giá kết quả thử nghiệm.

Do thời gian thực hiện đồ án cũng như kiến thức hiểu biết trong lĩnh vực này có hạn, vì vậy đồ án còn nhiều hạn chế và thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của thầy cô để đồ án được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

SINH VIÊN THỰC HIỆN ĐỒ ÁN

Phan Hoàng Trung
Trần Thị Ngọc

CHƯƠNG 1. TỔNG QUAN VỀ AN TOÀN VỚI CÁC PHƯƠNG PHÁP XÁC THỰC

1.1. Giới thiệu về ứng dụng Web

Website là một tập hợp các trang web có thể là văn bản, hình ảnh, video, Flash... Thường chỉ nằm trong một tên miền hoặc tên miền phụ. Trang web được lưu trữ trên máy chủ web có thể truy cập thông qua Internet.

Website có thể được xây dựng từ các thẻ HTML (website tĩnh) hoặc vận hành bằng các CMS chạy trên máy chủ (Website động). Website có thể được xây dựng bằng nhiều ngôn ngữ lập trình khác nhau (PHP, .NET, Java, Ruby on Rails...)

1.1.1. Khái niệm

Ứng dụng là một loại chương trình có khả năng làm cho máy tính thực hiện trực tiếp một công việc nào đó người dùng muốn thực hiện

Ban đầu, các website chỉ bao gồm text, hình ảnh và video, liên kết với nhau thông qua các link. Tác dụng của website là lưu trữ và hiển thị thông tin. Người dùng chỉ có thể đọc, xem, click các link để di chuyển giữa các page.

Về sau, với sự ra đời của các ngôn ngữ server: PHP, C#, Java, ... các website đã trở nên “động” hơn, có thể tương tác với người dùng. Từ đây, người dùng có thể dùng web để “thực hiện một chức năng nào đó mà người dùng mong muốn“, do đó web app ra đời.

Ứng dụng web chạy trên nền tảng website để chạy những phần mềm phục vụ nhu cầu của người sử dụng. Thông qua những web app, người sử dụng sẽ thực hiện những công việc có thể kể đến chẳng hạn như gửi file, tập tin, hình ảnh, video, mua hàng, thanh toán, gửi email hoặc thực hiện một truy vấn nào đó... Những thứ đã thường nghe hằng ngày như website bán hàng, website quản lý hàng hóa, website chia sẻ hình ảnh, website học trực tuyến... Thực chất chính là ứng dụng web.

Một web app tốt, thành công phải đảm bảo hệ thống được phân tích một cách tối ưu nhất có thể để đảm bảo hiệu suất và tính an toàn của ứng dụng web, nếu việc thiết kế website thông thường có thể tạo hình cơ bản trong vài ngày thì thiết kế web App phải trải qua một thời gian dài để phân tích hệ thống, xây dựng mô hình cơ sở dữ liệu một cách tỉ mỉ. Như vậy để thiết kế web đẹp có ứng dụng

tốt đòi hỏi lập trình viên phải có trình độ chuyên môn cao, có khả năng phân tích hệ thống bài bản, chuyên nghiệp.

Với sự phát triển của công nghệ thông tin thì việc phát triển các web app là một yêu cầu cần thiết và quan trọng với mỗi doanh nghiệp.

❖ Ưu điểm của web app

- Web app có khả năng bảo trì và cập nhật cao mà bạn không cần cài đặt trên các hệ thống máy tính. Đây là lý do tại sao web app được nhiều doanh nghiệp và lập trình viên lựa chọn mặc dù yêu cầu kỹ thuật cao hơn.
- Với web app bạn giúp công việc của các doanh nghiệp hiệu quả hơn, nhất là trong lĩnh vực hoạt động nhóm.
- Tính bảo mật của web app rất cao, đặc biệt là trong vấn đề sao lưu dữ liệu do web app có thể sử dụng công nghệ đám mây.
- Với web app thông tin của bạn sẽ được phủ sóng mọi nơi, phá vỡ khoảng cách địa lý so với các website thông thường.
- Khả năng ứng dụng rộng lớn: web app được sử dụng để làm các Webmail, đấu giá trực tuyến, bán hàng trực tuyến, diễn đàn thảo luận, wiki, MMORPG, Weblog, hệ quản trị nội dung, phần mềm quản lý nhân lực...
- Hoạt động tốt trên nhiều hệ điều hành khác nhau như: Mac OS, Windows, Linux, GNU...

❖ Nhược điểm của web app

- Thời gian phân tích nghiệp vụ lâu
- Tốn thời gian để xây dựng, phát triển
- Những rào cản về ngôn ngữ khi triển khai ứng dụng web
- Xu hướng công nghệ thay đổi nhanh
- Việc tối ưu, đảm bảo an toàn, tăng hiệu năng ... cần nghiên cứu kỹ lưỡng hoặc cần chuyên gia để giải quyết

1.1.2. Một số hướng phát triển ứng dụng Web

❖ Ứng dụng web tĩnh

Một ứng dụng web tĩnh, điều đầu tiên cần biết đó là loại ứng dụng web này sẽ hiển thị nội dung cố định muốn thay đổi cần phải chỉnh sửa file và cập nhật lên web server và đặc biệt là sẽ không có tính linh hoạt.

Thường thì người ta hay sử dụng HTML, CSS và Javascript để tạo nên những trang web như thế này. Tuy nhiên những đối tượng sinh động như banner, ảnh GIFs, video... có thể được bố trí thêm vào.

❖ Ứng dụng web động

So với ứng dụng web tĩnh thì ứng dụng web động phức tạp hơn về mặt kỹ thuật. Chúng sử dụng hệ cơ sở dữ liệu để load dữ liệu và nội dung được cập nhật mỗi khi người dùng truy cập, về cơ bản thì chúng đều có một nơi cho người có quyền quản trị web. Đây là nơi mà các quản trị viên có thể điều khiển và chỉnh sửa nội dung hiển thị, chức năng... của trang web.

Có rất nhiều ngôn ngữ lập trình khác nhau được sử dụng để lập trình một ứng dụng web động, trong đó PHP, ASP, JAVA, Python ... là những ngôn ngữ phổ biến nhất, tùy vào yêu cầu của khách hàng và điều kiện hiện tại của các tổ chức hoặc công ty sẽ lựa chọn ngôn ngữ để phát triển.

❖ Web thương mại điện tử

Nếu ứng dụng web là một cửa hàng online hoặc shop, việc lập trình sẽ là sự tái cấu trúc của một trang m-commerce hoặc e-commerce. Quá trình của kiểu ứng dụng này sẽ phức tạp hơn vì nó sẽ phải tích hợp cổng thanh toán điện tử qua credit card, Paypal... Lập trình viên đồng thời cũng phải thiết kế một panel quản lý cho admin bao gồm danh sách sản phẩm, thêm bớt và quản lý đơn hàng...

❖ Portal web app

Web portal hay còn gọi là cổng thông tin điện tử là một điểm truy cập tập trung và duy nhất, có thể tích hợp được các thông tin, ứng dụng, dịch vụ từ những nguồn khác nhau để chuyển đến người dùng thông qua một phương thức thống nhất trên nền tảng web.

Phân loại web portal dựa vào nhu cầu sử dụng của web portal, chúng ta có thể phân loại chúng thành các nhóm như sau:

- Web portal công cộng: là loại cổng thông tin được dùng để tổng hợp các thông tin lại từ nhiều nguồn, nhiều người khác nhau. Hoạt động giống trang yahoo. Web portal công cộng cho phép cá nhân hóa các website tùy vào sở thích của người sử dụng.
- Web portal doanh nghiệp: Là loại cổng thông tin được xây dựng dành cho các thành viên của doanh nghiệp sử dụng để phục vụ cho các nghiệp vụ của doanh nghiệp.

- Web portal giao dịch: Là cổng thông tin điện tử liên kết giữa người mua và người bán giống như ebay và chemweb.
- Web portal ứng dụng chuyên biệt: Là loại cổng thông tin được xây dựng cho các ứng dụng chuyên biệt khác nhau như SAP portal.

❖ Ứng dụng web hoạt hình, game

Hoạt hình ở đây được cấu thành từ công nghệ Flash, cho phép trang web hiển thị nội dung dưới dạng hoạt hình, đồng thời cũng cho phép việc thiết kế trở nên hiện đại và trực quan hơn. Đây cũng là một trong những công nghệ được sử dụng rộng rãi nhất bởi các nhà sáng tạo. Hạn chế của công nghệ này đó là không phù hợp với các công cụ SEO hay định vị web vì các engine tìm kiếm không thể đọc chính xác được nội dung mà trang web truyền tải.

❖ Ứng dụng web với hệ thống quản lý nội dung

Content Management System(CMS), hay còn gọi là hệ thống quản trị nội dung nhằm mục đích giúp dễ dàng quản lý, chỉnh sửa nội dung. Nội dung ở đây có thể là tin tức điện tử, báo chí hay các media hình ảnh, video, ... Hệ thống CMS giúp tiết kiệm thời gian quản lý, chi phí vận hành và bảo trì nên hiện nay có rất nhiều công ty sử dụng. Không chỉ là công ty mà hiện nay các blog cá nhân cũng ra đời với số lượng không hề nhỏ và họ chọn giải pháp sử dụng CMS nhằm dễ dàng xây dựng website và quản lý nội dung, bên cạnh đó còn tiết kiệm được chi phí xây dựng website. Chức năng chính của CMS là:

- Tạo, lưu trữ nội dung
- Chỉnh sửa nội dung
- Chuyển tải và chia sẻ nội dung
- Tìm kiếm và phân quyền người dùng
- Có rất nhiều CMS trên thế giới như: DotNetNuke(ASP), Drupal(PHP), Joomla(PHP), Wordpress(PHP), Kentiko(ASP)...

1.2. Tổng quan về xác thực

1.2.1. Một số khái niệm trong xác thực

Xác thực (Authentication) là việc xác lập hoặc chứng thực một thực thể (người nào đó hay một cái gì đó) đáng tin cậy, có nghĩa là những thông tin do một người đưa ra về một cái gì đó là đúng đắn. Xác thực một đối tượng có nghĩa là công nhận nguồn gốc của đối tượng, còn xác thực một người thường bao gồm

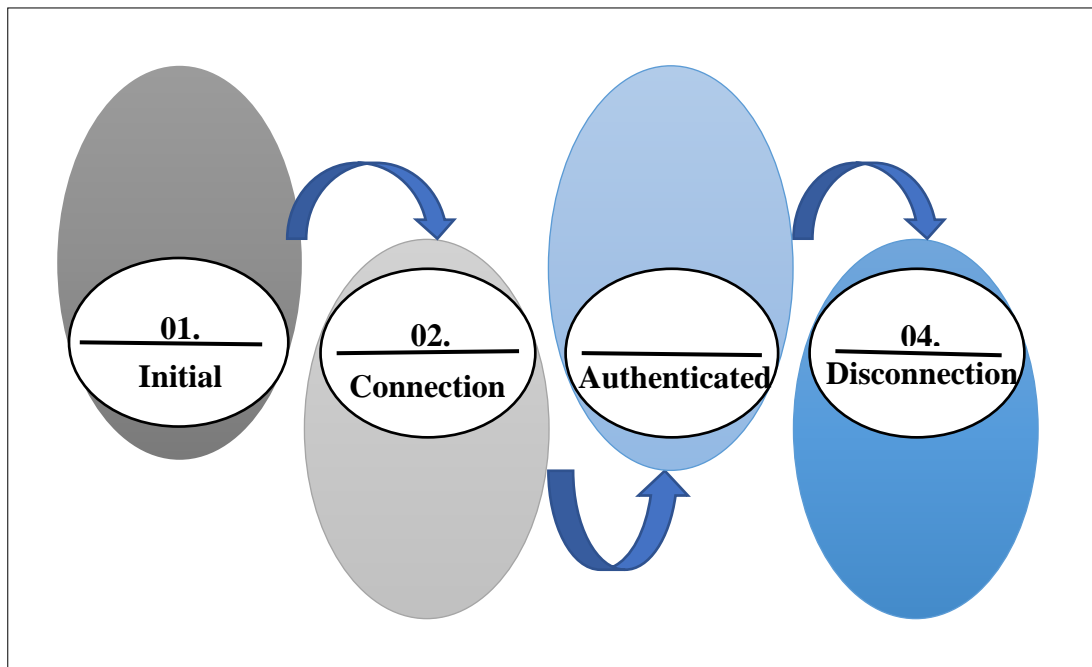
việc thẩm tra nhận dạng cá nhân của họ. Việc xác thực thường phụ thuộc vào một hoặc nhiều yếu tố xác thực (authentication factor) làm minh chứng cụ thể.

Xác thực là khâu đặc biệt quan trọng để bảo đảm an toàn cho hoạt động của một hệ thống thông tin. Đó là một quy trình nhằm xác minh nhận dạng số của bên gửi thông tin trong liên lạc trao đổi, xử lý thông tin, chẳng hạn như một yêu cầu đăng nhập. Bên gửi cần phải xác thực có thể là một người sử dụng máy tính, bản thân một máy tính hoặc một phần mềm. Đầu tiên, hệ thống luôn xác thực một thực thể khi nó cố gắng thử thiết lập liên lạc. Khi đó, nét nhận dạng của thực thể được dùng để xác định sự truy nhập của thực thể đó như một đặc quyền hoặc để đạt được sự sẵn sàng phục vụ. Đối với các giao dịch ngân hàng điện tử điển hình như giao dịch qua ATM/POS, giao dịch Online/Internet Banking, giao dịch Mobile Banking... thì xác thực là bắt buộc trong quản lý truy cập.

Xác thực thông điệp là một cơ chế cho phép khẳng định rằng thông điệp không bị thay đổi trong quá trình truyền và bên nhận có thể kiểm tra được nguồn gốc của thông điệp.

Xác thực thực thể là một thủ tục mà qua đó, một thực thể thiết lập một tính chất được yêu cầu cho một thực thể khác.

1.2.2. Các bước cơ bản trong xác thực



Hình 1.1. Mô hình các bước xác thực

❖ Một quá trình xác thực có thể chia thành 4 bước cơ bản.

- Bước đầu tiên trong quá trình xác thực: Bên được xác thực chưa được xác thực. Nó là thực thể thực hiện xác thực với hệ thống để sử dụng các dịch vụ. Nó có thể là một người hoặc một IS.
- Kết nối: Bên được xác thực sẽ gửi yêu cầu đến IS và yêu cầu xác thực, IS sử dụng chức năng yêu cầu xác thực để yêu cầu bộ kiểm tra xác thực yêu cầu đó. Bộ kiểm tra là thực thể cung cấp các dịch vụ xác thực. Nó xác nhận danh tính của bên được xác thực (hoặc từ chối trong trường hợp xác thực sai) và kiểm tra xem có thể cho phép việc sử dụng dịch vụ được yêu cầu hay không.
- Xác thực: Xác thực thành công, một phiên làm việc được mở, IS cung cấp các dịch vụ mà bên được xác thực yêu cầu.
- Ngắt kết nối: Người dùng ngắt kết nối hoặc bị ngắt kết nối bởi bộ kiểm tra và trạng thái trở lại bước ban đầu. Bước này có thể được bắt đầu sau một khoảng thời gian chờ đợi hoặc bởi một hành động của người dùng.

IS có thể yêu cầu các cấp độ xác thực khác nhau: cấp độ cho quản trị viên và cấp độ cho người dùng. Trong một hệ thống như vậy, mức độ xác thực được

chia theo thang điểm: mức 0 cho người dùng chưa được xác thực với quyền thấp nhất trong hệ thống, mức N cho người quản trị toàn quyền và một hoặc nhiều mức từ 0 đến N.

Tính bảo mật của một phương pháp xác thực phụ thuộc vào tính khả dụng và khả năng chấp nhận. Nếu tính khả dụng kém, người dùng sẽ nhanh chóng tìm cách bỏ qua các bước xác thực để sử dụng thuận tiện hơn. Điều này được xem là một nhược điểm.

Quá trình xác thực có thể dựa trên sự kết hợp của một hoặc nhiều yếu tố xác thực. Việc sử dụng nhiều hơn một yếu tố để xác thực được coi là xác thực mạnh.

1.3. Các yếu tố xác thực

Những cái mà người sử dụng sở hữu bẩm sinh, chẳng hạn như dấu vân tay, võng mạc mắt, chuỗi ADN, giọng nói, chữ ký, tín hiệu sinh điện đặc thù do cơ thể sống tạo ra hoặc những định dạng sinh trắc học khác.

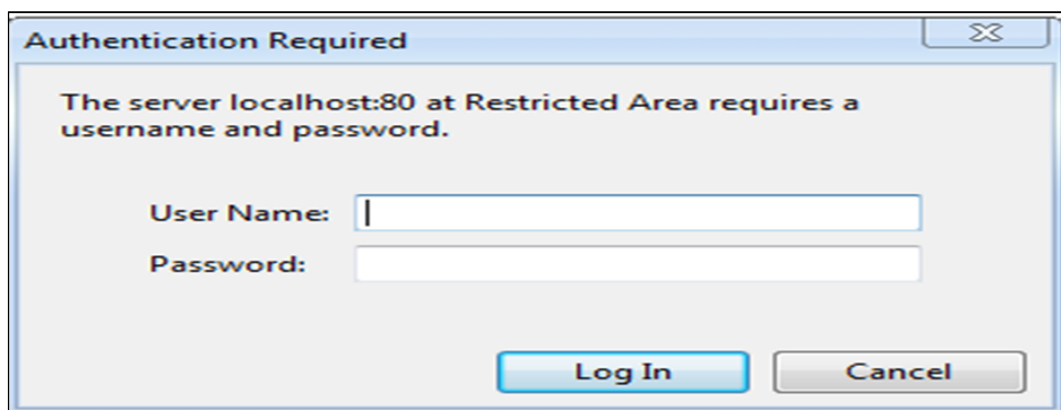
Những cái người sử dụng có, chẳng hạn như chứng minh thư, chứng chỉ an ninh (security token), chứng chỉ phần mềm (software token) hoặc điện thoại di động.

Những gì người sử dụng biết, chẳng hạn như mật khẩu (password), mật ngữ (pass phrase) hoặc mã số định danh cá nhân (personal identification number-PIN).

Trong thực tế, nhiều khi một tổ hợp của những yếu tố trên được sử dụng, lúc đó người ta nói đến xác thực đa yếu tố. Chẳng hạn trong giao dịch ATM, thẻ ngân hàng và mã số định danh cá nhân (PIN) được sử dụng – trong trường hợp này là một trong các dạng xác thực hai yếu tố (2FA).

1.4. Các phương pháp xác thực phổ biến

1.4.1. Phương pháp xác thực dựa trên định danh người dùng và mật khẩu



Hình 1.2. Form xác thực người dùng và mật khẩu

Đây là phương pháp xác thực truyền thống và được sử dụng phổ biến nhất hiện nay. Mỗi tài khoản sẽ bao gồm tên truy nhập và mật khẩu. Tên truy nhập dùng để phân biệt các người dùng khác nhau trong hệ thống (thường là duy nhất), còn mật khẩu để xác thực lại người sử dụng tên đó đúng là người sử dụng thật không. Mật khẩu thường do người sở hữu tên truy nhập tương ứng đặt và được giữ bí mật chỉ có người đó biết.

Khi người dùng muốn đăng nhập và sử dụng tài nguyên của hệ thống thì phải đăng nhập bằng cách nhập tên và mật khẩu của mình. Đầu tiên, hệ thống sẽ đối chiếu tên truy nhập của người dùng nhập vào với cơ sở dữ liệu tên người dùng, nếu tồn tại tên người dùng như vậy thì hệ thống tiếp tục đối chiếu mật khẩu được đưa vào tương ứng với tên truy nhập trong cơ sở dữ liệu. Qua 2 lần đối chiếu nếu thỏa mãn thì người dùng được xác thực và là người dùng hợp lệ của hệ thống, còn nếu không thì người dùng sẽ bị từ chối hoặc cấm truy cập.

❖ Ưu điểm và nhược điểm

– Ưu điểm

- Thiết kế và sử dụng đơn giản, tốn ít tài nguyên. Hệ thống chỉ gồm một cơ sở dữ liệu người dùng với 2 thông tin chủ yếu là tên truy nhập và mật khẩu. Tương ứng với mỗi tên truy nhập là quyền sử dụng của người đó trong hệ thống. Do đó các thông tin này không chiếm nhiều tài nguyên. Người dùng dễ hiểu và dễ sử dụng.
- Chi phí để thực hiện giải pháp này là rẻ so với các giải pháp khác. Nó không phụ thuộc vào các thiết bị phần cứng mà chỉ dựa trên phần mềm. Giải pháp này có khả năng làm việc trên mọi hệ điều

hành. Do đó, việc thực hiện giải pháp này khá dễ dàng và ít tốn kém.

– Nhược điểm

- Giải pháp này có nhược điểm lớn nhất là không có được sự bảo mật cao. Vì người dùng thường có tên đăng nhập mà nhiều người dùng sử dụng. Mặt khác, thông tin cặp Username và Password dùng để đăng nhập vào hệ thống mà ta gửi đi xác thực ở trong tình trạng ký tự văn bản rõ, tức không được mã hóa, có thể bị chặn bắt trên đường truyền và thường người dùng sẽ chọn mật khẩu dễ nhớ hoặc không cẩn thận khi gõ mật khẩu, do vậy dễ bị tấn công. Kẻ tấn công có nhiều phương pháp để đánh cắp được mật khẩu như thâm nhập vào hệ thống đọc file mật khẩu, dự đoán mật khẩu, vét cạn các từ trong từ điển để tìm mật khẩu hoặc người dùng có thể thiếu cẩn thận để lộ mật khẩu cho người khác.

❖ Một số giải pháp để tăng tính bảo mật cho phương pháp này:

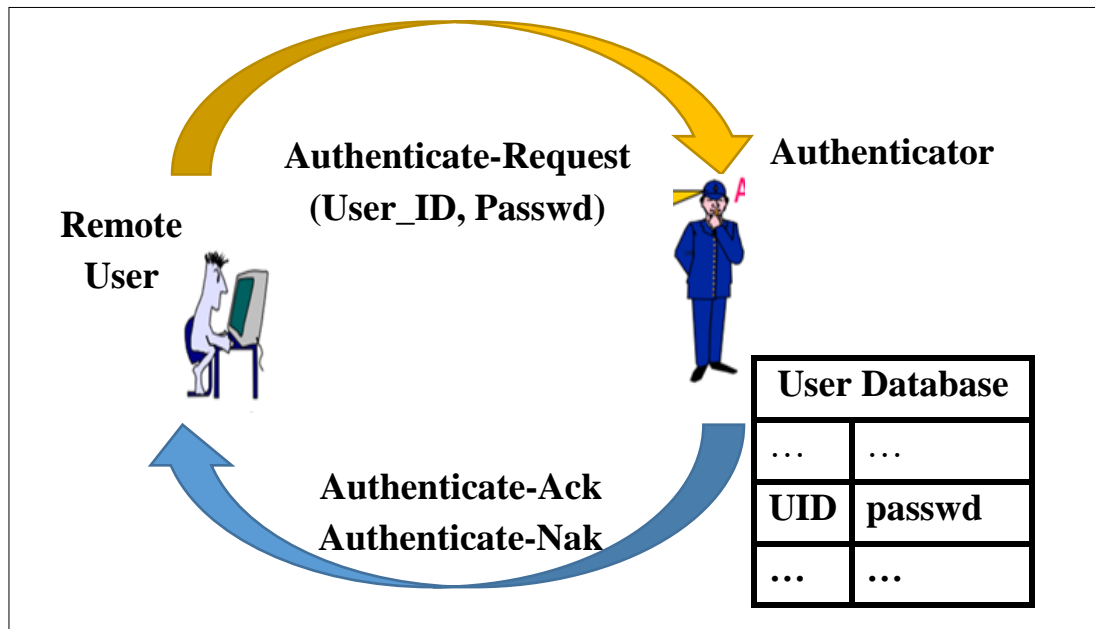
- Đặt mật khẩu phức tạp, mạnh: Mật khẩu phải chứa tối thiểu 8 ký tự, không trùng với tên đăng nhập, chứa các loại ký tự là chữ cái, chữ số, ký tự đặc biệt. Nếu đặt mật khẩu như vậy thì những kẻ muốn tấn công sẽ rất khó đoán được mật khẩu.
- Thay đổi mật khẩu: Quy định sau một thời gian nhất định mật khẩu sẽ không còn tác dụng đối với hệ thống và người dùng phải đặt lại mật khẩu khác. Mật khẩu sẽ được thay đổi nên khả năng kiểm soát tình trạng an toàn của mật khẩu cao hơn.
- Mã hóa thông tin: Trong môi trường làm việc là môi trường mạng, những nhà thiết kế thường dùng biện pháp mã hóa thông tin đăng nhập từ một máy khách nào đó trước khi chúng được gửi đi tới máy chủ của hệ thống. Do đó, khả năng mất cắp mật khẩu sẽ giảm đi rất nhiều khi kẻ xấu bắt được gói tin trên đường truyền.

Hiện nay, giải pháp mật khẩu sử dụng một lần (one-time password) được sử dụng rất nhiều trong các ứng dụng. Các mật khẩu trong danh sách chỉ có thể sử dụng một lần duy nhất mà không thể sử dụng lại trong những lần đăng nhập sau. Có 2 cách để hệ thống mật khẩu sử dụng một lần có thể làm việc là:

- Danh sách các mật khẩu được tạo ra một cách ngẫu nhiên bởi hệ thống và được sao làm 2 bản, một bản cho người dùng và một bản cho hệ thống.
- Danh sách mật khẩu được tạo ra theo yêu cầu của người sử dụng và được hệ thống công nhận.

1.4.2. Phương pháp xác thực sử dụng giao thức PAP.

Giao thức xác thực PAP là một giao thức bắt tay hai chiều. Đó là máy tính tạo kết nối và gửi nhận dạng người dùng, mật khẩu kép đến hệ thống đích mà nó cố gắng thiết lập một kết nối và sau đó hệ thống đích xác thực rằng máy tính đó được xác thực đúng và được chấp nhận cho việc truyền thông. Xác thực PAP có thể được dùng khi bắt đầu của kết nối PPP, cũng như trong suốt một phiên làm việc của PPP để xác thực kết nối.



Hình 1.3. Giao thức PAP

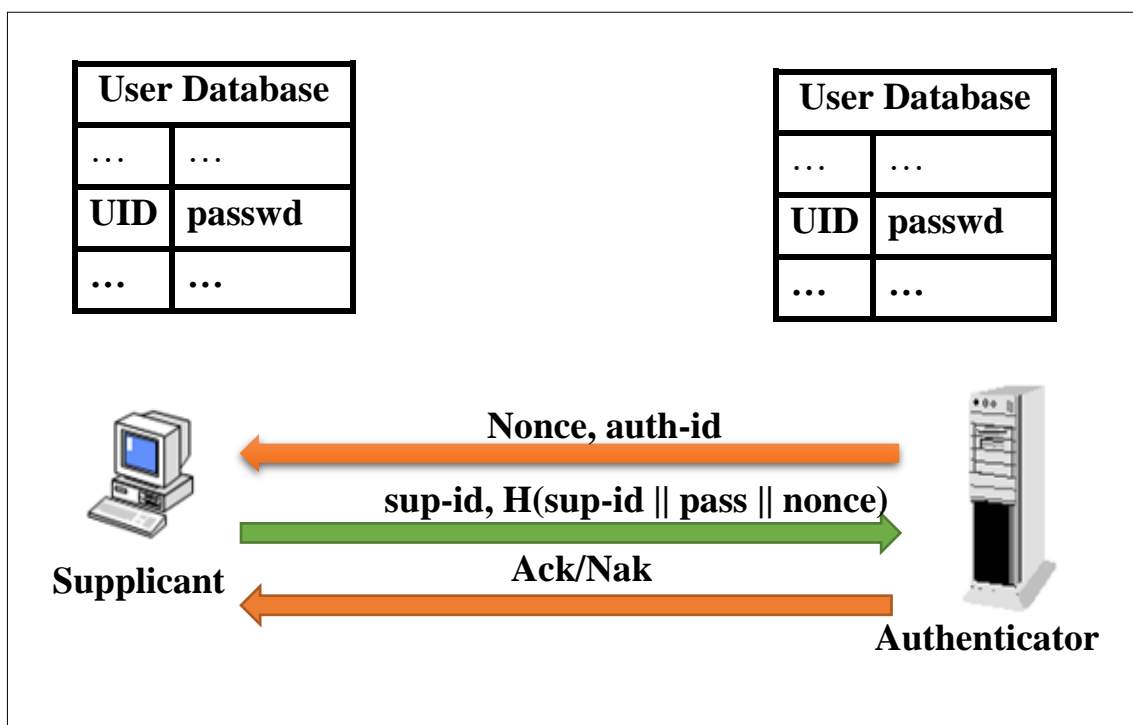
❖ Hoạt động của giao thức PAP

- Bước 1: User sẽ gửi một Authenticate-Request gồm User_ID, Passwd đến Authenticator. Authenticator sẽ kiểm tra User_ID được truyền đến có tồn tại trong cơ sở dữ liệu không, nếu có nó sẽ đọc ra mật khẩu tương ứng và đem so sánh với mật khẩu được gửi đến.
- Bước 2: Trả về thông điệp phản hồi Authenticate-Ack nếu thành công và Authenticate-Nak nếu thất bại.

PAP không bảo mật vì thông tin xác thực được truyền đi dạng rõ và dễ bị chặn thu trên đường truyền.

1.4.3. Phương pháp xác thực sử dụng giao thức CHAP.

Là giao thức bắt tay ba bước, xác thực sử dụng mật khẩu, không truyền mật khẩu dạng rõ



Hình 1.4. Giao thức CHAP

❖ Hoạt động của giao thức CHAP

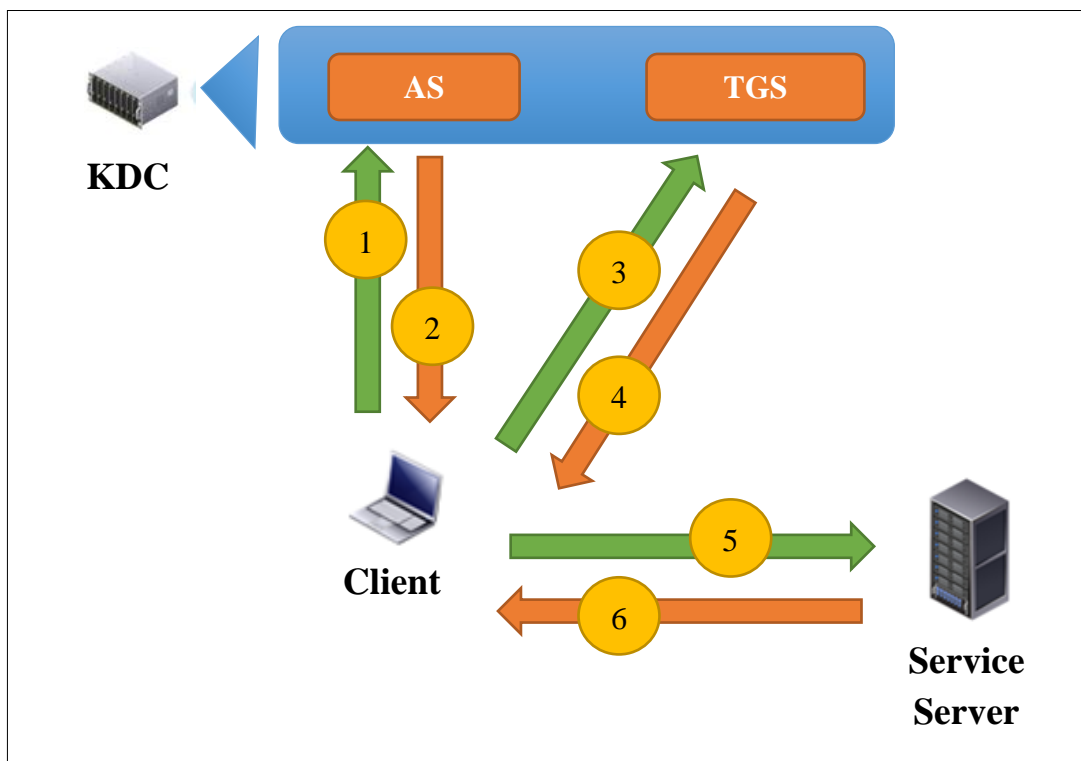
- Bước 1: Authenticator bắt đầu quá trình xác thực bằng cách gửi nonce, auth-id đến supplicant.
- Bước 2: Supplicant sẽ căn cứ vào auth-id xác định được passwd để giải đố trong cơ sở dữ liệu và gửi lại cho Authenticator gồm sup-id, $H(\text{sup-id} \parallel \text{passwd} \parallel \text{nonce})$. Khi đó Authenticator nhận được, căn cứ vào sup-id để xác định passwd trong cơ sở dữ liệu, tiến hành băm và so sánh với giá trị băm nhận được của Supplicant.
- Bước 3: Trả về Ack nếu thành công và Nak nếu thất bại.

Phương thức CHAP thường được sử dụng khi người dùng đăng nhập vào các máy chủ ở xa (remote server) của hệ thống, chẳng hạn như RAS server. Dữ liệu chứa mật khẩu được mã hóa đôi khi được gọi là “mật khẩu băm” (hash password) theo tên của phương pháp mã hoá dùng các hàm băm.

1.4.4. Phương pháp xác thực sử dụng giao thức Kerberos.

Là nền tảng xác thực chính của nhiều hệ điều hành như UNIX, Windows. Xây dựng dựa trên giao thức Needham-Schroeder, sử dụng mật mã đối xứng và có bên thứ ba tin cậy là “Trung tâm phân phối khóa”.

Xác thực Kerberos dùng một máy chủ trung tâm để kiểm tra việc xác thực người dùng và cấp phát thẻ dịch vụ (service ticket) để người dùng có thể truy cập vào tài nguyên hệ thống. Xác thực Kerberos là một phương thức có tính an toàn cao nhờ việc dùng thuật toán mã hóa mạnh. Kerberos cũng dựa trên độ chính xác của thời gian xác thực giữa máy chủ và người dùng, do đó cần phải đảm bảo kết nối đồng bộ thời gian giữa các thành phần này của hệ thống.



Hình 1.5. Giao thức Kerberos

❖ Hoạt động của giao thức Kerberos

- Bước 1: Client truy cập đến AS, yêu cầu được truy cập vào hệ thống.
- Bước 2: Khi đó AS sẽ xác thực và cấp TGT.
- Bước 3: Client sử dụng TGT để truy cập vào TGS để xin một ST.
- Bước 4: TGS kiểm tra yêu cầu truy cập dịch vụ hợp lệ, sẽ cấp cho Client một ST.
- Bước 5: Client sử dụng ST để truy cập tới Service Server và yêu cầu phục vụ.
- Bước 6: Service Server kiểm tra tính hợp lệ và đáp ứng yêu cầu.

1.4.5. Phương pháp xác thực sử dụng Tokens

Là phương tiện vật lý như các thẻ thông minh (smart card) hoặc thẻ đeo của nhân viên (ID badges) chứa thông tin xác thực. Tokens có thể lưu trữ số nhận dạng cá nhân – personal identification number (PINs), thông tin về người dùng hoặc mật khẩu.

Các thông tin trên token chỉ có thể được đọc và xử lý với các thiết bị đặc dụng, ví dụ như thẻ smart card được đọc bởi đầu đọc smart card gắn trên máy tính, sau đó thông tin này được gửi đến “authenticating server”. Tokens chứa chuỗi text hoặc giá trị số duy nhất, thông thường mỗi giá trị này chỉ sử dụng một lần.



Hình 1.6. Thẻ thông minh

Thẻ thông minh (smart card) là một thẻ plastic có kích cỡ như thẻ tín dụng được trang bị một vi mạch dùng để chứa bộ nhớ và một mạch xử lý với hệ điều hành để kiểm soát bộ nhớ.

Nó có thể lưu trữ dữ liệu về thông tin các nhân, tiền hoặc một số thông tin khác mà sự thay đổi của chúng cần được kiểm soát chặt chẽ. Ngoài ra, nó có thể lưu trữ các khóa mã hóa để người dùng có thể nhận dạng qua mạng, chữ ký điện tử. Đặc biệt là thẻ thông minh có hỗ trợ chứng nhận số. Nó mã hóa dữ liệu và kiểm tra tính hợp lệ của các giao dịch qua mạng. Đây là một giải pháp rất hiệu quả và linh động cho các vấn đề về xác thực người dùng.

Hiện nay, các cơ quan tổ chức dùng thẻ rất nhiều. Đầu tiên, những thông tin cần thiết cho việc nhận dạng các nhân viên trong cơ quan, tổ chức sẽ được lưu vào bộ nhớ của thẻ. Sau đó, nó được cung cấp cho các nhân viên tương ứng với các thông tin đó. Mỗi cơ quan, tổ chức khác nhau sẽ có các yêu cầu về thông

tin xác thực khác nhau nhưng thường là các thông tin như tên truy nhập, mật khẩu và một số thông tin cá nhân khác.

Trong hệ thống thông tin đòi hỏi phải có xác thực người dùng, nhân viên trong tổ chức chỉ cần đưa thẻ, thiết bị đọc thẻ và nhập vào một mã số bí mật nào đó để xác nhận với hệ thống là chính họ là người sở hữu chiếc thẻ đó. Khi đã nhập đúng mã này, thiết bị đọc thẻ sẽ đọc các thông tin nhận dạng được ghi trong thẻ và chuyển các thông tin này vào hệ thống, sau đó hệ thống sẽ kiểm tra chúng với cơ sở dữ liệu người dùng.

❖ Ưu điểm và nhược điểm

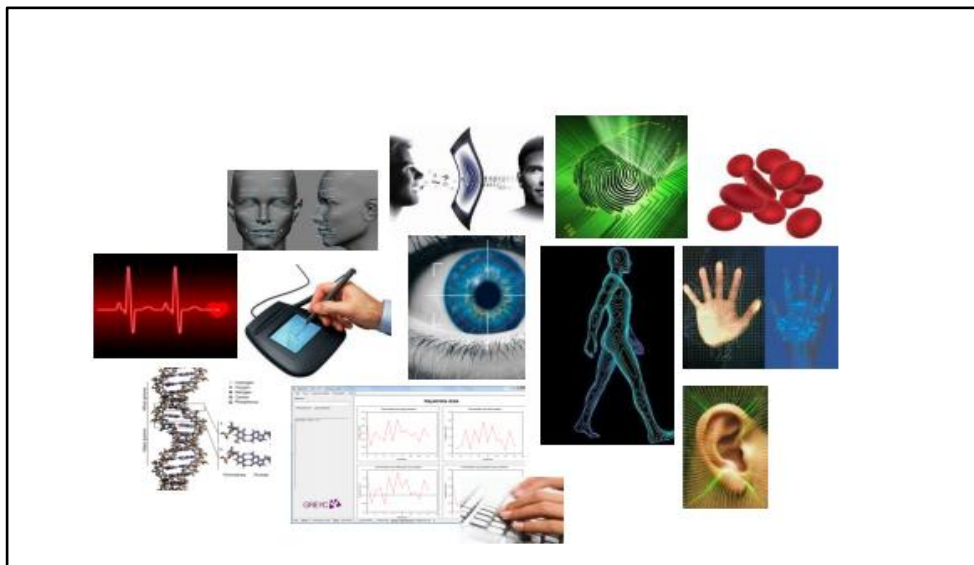
– Ưu điểm

- Nhờ vào kiến trúc vật lý và logic của thẻ mà đã giảm được rất nhiều các nguy cơ gây mất an toàn thông tin. Mọi hoạt động của thẻ đều được kiểm soát bởi hệ điều hành nên các thông tin cần giữ bí mật sẽ không thể lấy ra từ thẻ. Các thông tin bên trong thẻ không thể bị kẻ xấu lấy cắp như các thông tin được lưu trữ trong các phần mềm hệ quản trị cơ sở dữ liệu thông thường.
- Các khóa bí mật dùng cho chữ ký điện tử và nhận dạng đều được lưu trữ bên trong thẻ. Nhà sản xuất thẻ cũng như người sở hữu thẻ đều không thể biết được các khóa này. Vì vậy, chúng không thể bị sao chép hay bị lấy cắp.
- Mỗi chiếc thẻ đều có số nhận dạng PIN để tránh việc đánh cắp và bị kẻ xấu sử dụng. Trước khi sử dụng thẻ, người dùng phải nhập vào số PIN của thẻ. Cơ chế quản lý số PIN của thẻ cũng rất an toàn bởi vì số PIN gần như không thể đoán ra được. Mặt khác, thẻ quy định số lần nhập tối đa, nếu số lần nhập không chính xác liên tục đến con số quy định thì thẻ sẽ tự động khóa. Muốn mở khóa thì người dùng phải nhập vào một số dùng để mở khóa của thẻ. Tương tự, nếu nhập không chính xác liên tiếp đến một con số nào đó thì thẻ sẽ bị khóa vĩnh viễn và không thể sử dụng lại nữa. Như vậy, việc sử dụng thẻ rất an toàn và thuận tiện. Giờ đây người dùng thay vì phải nhớ nhiều số thì chỉ phải nhớ một số, còn các thông tin nhận dạng đều ở trong thẻ. Trong trường hợp thẻ bị mất cắp, kẻ lấy cắp cũng không thể sử dụng được thẻ vì không có số PIN.

– Nhược điểm

- Tuy giải pháp này đã hạn chế được sự mất cắp thẻ bằng cách kết hợp thẻ với một số PIN nhưng vẫn có thể bị đánh cắp cả thẻ và số PIN. Bất người dùng phải nhớ số PIN và phải thêm một chiếc thẻ mới mới có thể thực hiện việc xác thực.
- Để áp dụng giải pháp này, các cơ quan phải trang bị thêm các thiết bị như thiết bị đọc thẻ, thiết bị ghi, các phần mềm hỗ trợ. Số lượng và giá thành của các thiết bị này không phải là nhỏ, do đó khá là tốn kém.
- Các dịch vụ hỗ trợ phổ biến cho việc xác thực bằng thẻ là chưa đầy đủ. Các dịch vụ thư điện tử, các dịch vụ thương mại cần đến xác thực trên Internet đều chưa hỗ trợ xác thực bằng thẻ. Hiện nay, hầu như các nhà cung cấp giải pháp xác thực người dùng bằng thẻ đều phát triển các dịch vụ theo mô hình riêng của mình, sử dụng các thiết bị riêng chưa thống nhất, do đó khả năng tương thích giữa các hệ thống hầu như không có.

1.4.6. Xác thực bằng sinh trắc học



Hình 1.7. Xác thực sinh trắc học

Tuy giải pháp xác thực sử dụng thẻ thông minh khá an toàn và linh hoạt nhưng trong những lĩnh vực quan trọng cần an toàn chặt chẽ như ngân hàng, quân sự thì vẫn đòi hỏi phải có giải pháp khác an toàn hơn. Vì thế các nhà nghiên cứu đã đưa ra giải pháp xác thực sử dụng những kỹ thuật sinh trắc học để giải quyết những vấn đề đó.

Sinh trắc học là công nghệ sử dụng những thuộc tính vật lý, đặc điểm sinh học riêng của mỗi cá nhân như vân tay, khuôn mặt, móng mắt, tĩnh mạch để nhận diện, xác thực bảo mật.

Xác thực sinh trắc học là một hình thức bảo mật đo lường và so sánh các tính năng sinh trắc học của người dùng để xác minh rằng một người được phép truy cập vào một thiết bị cụ thể.

❖ Đặc điểm

- Các đặc điểm vật lý và sinh học dành riêng cho một cá nhân và có thể dễ dàng so sánh với các đặc điểm được phép lưu trong cơ sở dữ liệu.
- Xác thực sinh trắc học cũng có thể được cài đặt trong môi trường vật lý và sử dụng để kiểm soát các điểm truy cập.

❖ Hoạt động

- Ở phía máy khách, người dùng sử dụng một thiết bị đầu cuối có hỗ trợ biểu mẫu dùng cho việc đăng nhập vào hệ thống hoặc trong môi trường Internet thì sử dụng trình duyệt để mở trang đăng nhập.
- Người dùng sẽ phải điền vào biểu mẫu mật khẩu hay một thông tin nhận dạng tương tự và cung cấp mẫu sinh trắc học như dấu vân tay, hình dạng lòng bàn tay, mắt, giọng nói, chữ ký. Thông qua các thiết bị nhận dạng được tích hợp trong đó. Sau đó, các thông tin này sẽ được chuyển về trung tâm xác thực của hệ thống để kiểm tra. Trung tâm sẽ phân tích mẫu thu được và đối chiếu xem mẫu tương ứng với mật khẩu được lưu trong cơ sở dữ liệu có trùng hay không, nếu trùng thì người dùng đăng nhập là hợp lệ và hệ thống sẽ đáp ứng các quyền hạn, tài nguyên phù hợp.

❖ Các phương pháp xác thực sinh trắc học phổ biến

- **Cảm biến vân tay:** Hiện nay chắc hẳn chúng ta đã không còn xa lạ gì với những chiếc điện thoại tích hợp cảm biến vân tay, các khóa cửa sử dụng dấu vân tay làm chìa khóa. Khi đặt ngón tay lên trên phần cảm biến vân tay, thiết bị sẽ quét hình ảnh ngón tay và đưa vào hệ thống xử lý. Hệ thống này chuyển dữ liệu vừa nhận sang dạng số rồi đối chiếu các đặc điểm của vân tay đó với dữ liệu đã được lưu trữ trong hệ thống. Nếu khớp thì thì lớp khóa hay bảo mật sẽ được mở. Mặc dù đôi khi không chính xác, nhưng đây là một

trong những công nghệ sinh trắc học phổ biến và được sử dụng nhiều nhất hiện nay.

- **Nhận diện giọng nói:** Các công nghệ nhận dạng giọng nói đo lường các đặc điểm của giọng nói để phân biệt giữa các cá nhân. Giống như máy quét khuôn mặt, chúng kết hợp một số điểm dữ liệu và tạo hồ sơ dấu vết giọng nói để so sánh với cơ sở dữ liệu. Thay vì nghe giọng nói, công nghệ nhận dạng giọng nói tập trung vào việc đo lường, kiểm tra miệng và cổ họng của người nói để hình thành các hình dạng, chất lượng âm thanh cụ thể. Quy trình này tránh các vấn đề bảo mật có thể gây ra do cố gắng nguy trang, bắt chước giọng nói hoặc do các tình trạng phổ biến như ốm đau, thời gian trong ngày có thể thay đổi chất lượng âm thanh của giọng nói đối với tai người. Những từ người dùng nói để truy cập vào một thiết bị được bảo vệ bằng giọng nói cũng có thể được tiêu chuẩn hóa một chút, đóng vai trò như một loại mặt khẩu và thực hiện so sánh các bản ghi âm được phê duyệt với người dùng.
- **Nhận diện qua mắt:** Cơ chế nhận diện mống mắt của thiết bị được thực hiện nhờ một máy chiếu bước sóng nhìn thấy được hoặc tia hồng ngoại tầm gần vào mắt người. Mục đích của việc chiếu tia đặc biệt này vào mắt nhằm giúp xác định chính xác vị trí của từng bộ phận của mắt: đồng tử, mí mắt, lông mi và chi tiết cấu trúc của mống mắt. Sau đó, mọi dữ liệu thu thập được sẽ được chuyển về thiết bị phần mềm để dò tìm các tham số như tần số, biên độ. Cuối cùng, dữ liệu này được mã hóa bằng các thuật toán đặc biệt và phép thống kê để đưa ra kết quả nhận diện chính xác chủ nhân của thiết bị. Nếu đúng, khóa sẽ được mở.
- **Nhận diện khuôn mặt:** Công nghệ nhận dạng khuôn mặt dựa trên việc so khớp hàng chục phép đo khác nhau từ khuôn mặt đã được phê duyệt với khuôn mặt của người dùng đang cố gắng truy cập, tạo ra cái được gọi là khuôn mặt. Tương tự như máy quét vân tay, nếu đủ số lượng phép đo từ người dùng khớp với khuôn mặt đã được phê duyệt, quyền truy cập sẽ được cấp.

❖ Ưu điểm và nhược điểm

- Ưu điểm

- Người dùng hầu như không thể thay đổi được các bộ phận như dấu vân tay, mắt...dùng trong xác thực.
 - Người dùng cũng không thể đưa những đặc điểm này cho người khác sử dụng như thẻ hay mặt khẩu được.
 - Các đặc điểm sinh trắc học này thì không thể bị mất cắp. Ngày nay với trình độ khoa học công nghệ phát triển, việc nhận biết các thông tin sinh trắc học đã có thể phân biệt được thông tin sinh trắc học của người sống và của người chết.
- Nhược điểm
- Khi mà các dữ liệu sinh trắc học khó có sự thay đổi như dấu vân tay hay mắt được sử dụng trong các ứng dụng khác nhau thì rất dễ bị đánh cắp.
 - Trên thế giới vẫn chưa có một chuẩn chung nào cho việc số hóa các mẫu sinh trắc học. Mặt khác, các nhà sản xuất khác nhau cung cấp các thiết bị xác thực mẫu sinh trắc học theo các chuẩn khác nhau, không có sự thống nhất. Do đó, việc trang bị hệ thống xác thực này không có tính linh động cao.
 - Có một số thông tin có thể bị thay đổi vì nhiều lý do. Ví dụ: Dấu vân tay bị thay đổi do bị chấn thương, giọng nói bị méo do bị viêm họng. Do đó, việc xác thực đúng các thông tin này thường rất thấp.
 - Ở nhiều nơi việc đưa giải pháp này vào các ứng dụng trên Internet là không thực tế. Các thông tin xác thực sinh trắc học thường khá lớn trong khi băng thông đường truyền không phải ở đâu cũng đủ rộng. Dẫn đến kết quả phản hồi lại rất chậm.

Cùng với sự phát triển và ứng dụng của công nghệ thông tin vào cuộc sống thì công nghệ xác thực bằng sinh trắc học đã và đang có những bước phát triển đáng kinh ngạc. Nhưng vì nhận dạng sinh trắc học hiện rất tốn kém chi phí khi triển khai nên chưa được sử dụng rộng rãi như các phương thức xác thực khác.

1.4.7. Phương pháp xác thực đa yếu tố

Xác thực đa yếu tố (Multi-Factor Authentication) là phương thức xác thực dựa trên nhiều yếu tố xác thực kết hợp, là mô hình xác thực yêu cầu kiểm chứng ít nhất là hai yếu tố xác thực. Phương thức này là sự kết hợp của bất cứ yếu tố xác thực nào, ví dụ như yếu tố đặc tính sinh trắc của người dùng

hoặc những gì người dùng biết để xác thực trong hệ thống.

Với xác thực đa yếu tố, ngân hàng có thể tăng mức độ an toàn, bảo mật cho giao dịch trực tuyến lên rất nhiều nhờ việc kiểm chứng nhiều yếu tố xác thực. Ví dụ như xác thực chủ thẻ trong giao dịch ATM, yếu tố xác thực đầu tiên của khách hàng là thẻ ATM (cái khách hàng có), sau khi đưa thẻ vào máy, khách hàng sẽ phải đưa tiếp yếu tố xác thực thứ hai là số PIN (cái khách hàng biết). Một ví dụ khác là xác thực người sử dụng dịch vụ giao dịch Internet Banking: khách hàng đăng nhập với Username và Password sau đó còn phải cung cấp tiếp OTP (One - Time - Password) được sinh ra trên token của riêng khách hàng.

An toàn, bảo mật trong giao dịch trực tuyến là hết sức quan trọng, trong đó xác thực người sử dụng là một trong những khâu cốt lõi. Với xác thực đa yếu tố, ta có thể tăng mức độ an toàn, bảo mật nhờ việc kiểm chứng nhiều yếu tố xác thực. Mức độ an toàn bảo mật sẽ càng cao khi số yếu tố xác thực càng nhiều. Khi số yếu tố xác thực lớn thì hệ thống càng phức tạp, kéo theo chi phí đầu tư và duy trì vận hành tốn kém, đồng thời lại bất tiện cho người sử dụng. Do vậy, trên thực tế để cân bằng giữa an toàn, bảo mật và tính tiện dụng, người ta thường áp dụng xác thực hai yếu tố và xác thực ba yếu tố (3FA).

Xác thực đa yếu tố dù có mức độ an toàn, bảo mật cao hơn, nhưng cũng cần các biện pháp nghiệp vụ khác để bảo đảm tuyệt đối an toàn trong các hoạt động giao dịch trực tuyến.

1.5. Tổng kết chương 1

Trong chương này tập trung vào tìm hiểu về định nghĩa, các bước cơ bản của một quá trình xác thực, các yếu tố xác thực và những phương pháp xác thực phổ biến được sử dụng hiện nay. Xác thực truyền thống được thực hiện bằng cách sử dụng mô hình máy chủ xác thực máy khách, trong đó máy khách sẽ yêu cầu tài nguyên được bảo vệ trên máy chủ bằng cách xác thực thông tin đăng nhập với chủ sở hữu tài nguyên. Trong mô hình này, cách duy nhất để chủ sở hữu tài nguyên cung cấp quyền cho ứng dụng bên thứ ba truy cập vào tài nguyên được bảo vệ là để chia sẻ thông tin đăng nhập của chủ sở hữu tài nguyên với bên thứ ba, việc chia sẻ như thế có nhiều vấn đề và hạn chế. Từ đó OAuth2 ra đời, nhằm mục đích giải quyết các vấn đề của mô hình máy khách-máy chủ bằng cách ủy quyền và tách biệt vai trò của máy khách với vai trò của chủ sở hữu tài nguyên. Chúng ta sẽ tìm hiểu nhiều hơn về OAuth2 ở chương tiếp theo.

CHƯƠNG 2. GIẢI PHÁP XÁC THỰC VỚI OAUTH2

2.1. Tổng quan về OAuth 2.0

2.1.1. Đặt vấn đề

Trong mô hình xác thực client-server truyền thống, máy client yêu cầu tài nguyên bị hạn chế quyền truy cập trên máy chủ bằng cách xác thực thông tin đăng nhập của chủ sở hữu tài nguyên với máy server. Để cung cấp cho các ứng dụng bên thứ ba quyền truy cập vào tài nguyên bị hạn chế, chủ sở hữu tài nguyên phải chia sẻ thông tin đăng nhập với bên thứ 3. Điều này tạo ra một số vấn đề và hạn chế:

- Các ứng dụng bên thứ ba được yêu cầu lưu trữ thông tin đăng nhập của chủ sở hữu để sử dụng trong tương lai, thường là mật khẩu ở dạng rõ.
- Máy chủ được yêu cầu hỗ trợ xác thực mật khẩu, mặc dù những điểm yếu trong bảo mật vốn có trong mật khẩu.
- Các ứng dụng bên thứ ba có quyền truy cập khá rộng vào tài nguyên được bảo vệ bởi chủ sở hữu, chủ sở hữu tài nguyên không có khả năng hạn chế thời lượng truy cập và phạm vi quyền truy cập.
- Chủ sở hữu tài nguyên không thể thu hồi quyền truy cập vào một bên thứ ba cá nhân mà không thu hồi quyền truy cập vào tất cả các bên thứ ba khác bằng cách thay đổi mật khẩu đăng nhập.
- Thỏa hiệp bất kỳ ứng dụng của bên thứ ba nào dẫn đến thỏa hiệp mật khẩu của người dung cuối, và tất cả dữ liệu của người dung cuối đều được bảo vệ bởi mật khẩu đó.

OAuth giải quyết tất cả các vấn đề này bằng cách giới thiệu một lớp ủy quyền tách biệt vai trò của khách hàng với vai trò của chủ sở hữu tài nguyên. Trong oauth máy client yêu cầu quyền truy cập vào các tài nguyên được kiểm soát bởi chủ sở hữu tài nguyên, được lưu trữ bởi máy chủ tài nguyên và ban hành một bộ thông tin xác thực khác với thông tin tài khoản chủ sở hữu tài nguyên.

Thay vì sở hữu thông tin đăng nhập của chủ sở hữu tài nguyên để truy cập vào tài nguyên, client có được mã thông báo truy cập(Access Token) có giới hạn về thời lượng và phạm vi truy cập cụ thể. Mã thông báo truy cập được cấp cho máy Client bởi máy chủ ủy 2 quyền với sự phê duyệt của chủ sở hữu tài

nguyên. Máy khách sử dụng sử dụng mã thông báo truy cập để truy cập vào các tài nguyên được lưu trữ bởi máy chủ tài nguyên.

2.1.2. Lịch sử phát triển

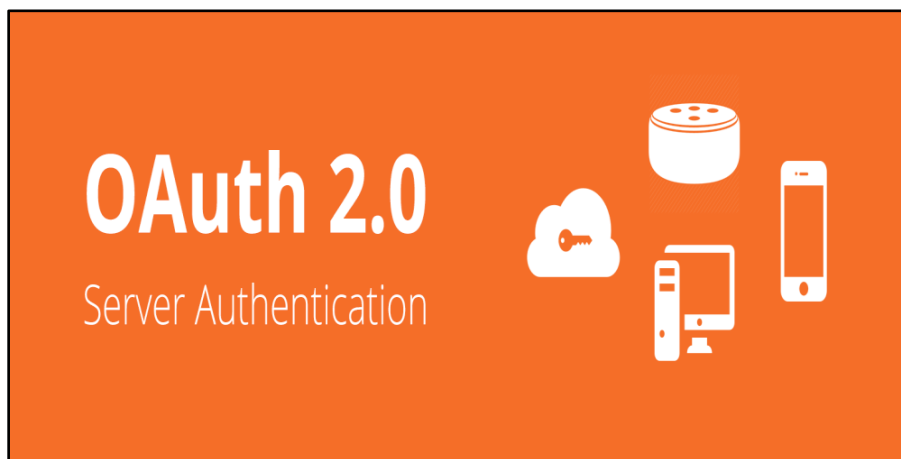
Về cơ bản, OAuth là một phương thức mà từ một phía máy chủ hay một bên thứ 3 có thể đại diện cho người dùng khai báo cũng như chứng thực người dùng để truy cập vào tài nguyên người dùng nằm trong một dịch vụ của một ứng dụng nào đó. OAuth cũng phát triển đồng hành với sự phát triển của các dịch vụ mạng xã hội như Facebook, Twitter, Instagram. Đặc biệt là khi các công ty như Twitter cung cấp API để bên thứ 3 có thể truy cập vào tài nguyên của người dùng, một thị trường đầy tiềm năng và mới mẻ mới đã mở ra. Đó là thị trường ứng dụng chia sẻ từ bên thứ 3.

Chúng ta cùng điểm lại vài nét trong lịch sử phát triển của OAuth2

- Năm 2006, Twitter phát triển hệ thống OpenID phục vụ cho đăng nhập các ứng dụng trong hệ thống của Twitter, tuy nhiên hệ thống này yêu cầu người dùng phải cung cấp tên đăng nhập và mật khẩu, đây là một điểm yếu. Cũng chính năm này, các ông lớn trong về mạng xã hội như Facebook, Google, Twitter... đã cùng ngồi với nhau để phác thảo ý tưởng về một hệ thống xác thực mới giúp các ứng dụng bên thứ ba có thể tích hợp.
- Năm 2008, IETF - tổ chức quản lý tiêu chuẩn mạng Internet (tránh nhầm với IEEE tổ chức quản lý tiêu chuẩn trong lĩnh vực điện và điện tử) đã quyết định hỗ trợ tiêu chuẩn OAuth này và bắt đầu cho xây dựng RFC 1.0.
- Năm 2010, IETF phát hành phiên bản chính thức đầu tiên của OAuth 1.0. Sau đó, lỗi bảo mật nghiêm trọng được phát hiện với tên gọi Session Fixation xảy ra trên OAuth 1.0, cho phép các hacker lừa ứng dụng bên thứ ba trao quyền truy nhập vào tài khoản và dữ liệu người dùng.
- Năm 2012, phiên bản OAuth 2.0 ra đời, tuy nhiên vẫn còn những lỗi bảo mật chết người các hacker có thể Hack Facebook thông qua trình duyệt Chrome nhưng vẫn đang được sử dụng khá rộng rãi. Từ đó đến nay chưa có thêm một phiên bản nào khác của OAuth ra đời thêm.

2.1.3. Giới thiệu và định nghĩa OAuth 2.0

OAuth2 là bản nâng cấp của OAuth1.0, là một giao thức chứng thực cho phép các ứng dụng chia sẻ một phần tài nguyên với nhau mà không cần xác thực qua username và password.



Hình 2.1. Giới thiệu về OAuth 2.0

OAuth2 là viết tắt của Open với Authentication hoặc Authorization, tức bao gồm cả hai công việc:

- Authentication: xác thực người dùng.
- Authorization: người dùng ủy quyền cho ứng dụng truy cập tài nguyên của họ.

OAuth2 không đơn thuần chỉ là giao thức kết nối, nó là một "nền tảng" mà chúng ta phải triển khai ở cả hai phía: Client và Server.

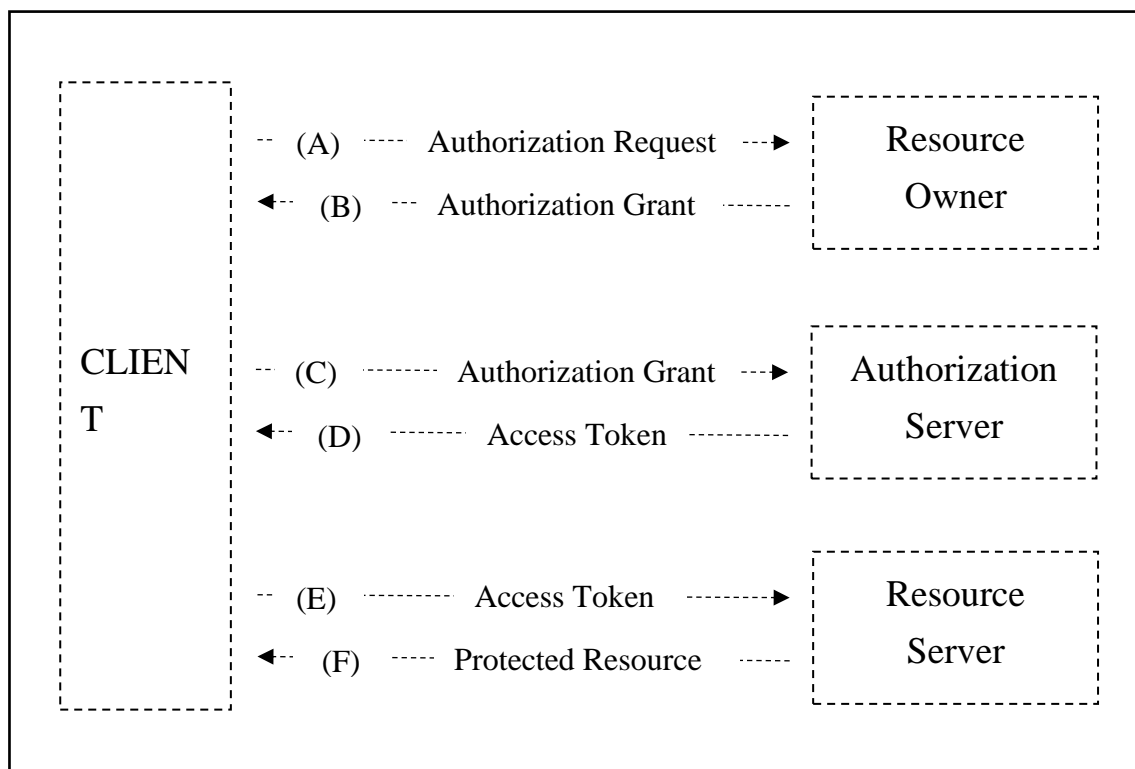
Các vai trò trong OAuth2:

- Resource owner: là những người dùng có khả năng cấp quyền truy cập, chủ sở hữu của tài nguyên mà ứng dụng muốn lấy.
- Resource server: nơi lưu trữ các tài nguyên, có khả năng xử lý yêu cầu truy cập đến các tài nguyên được bảo vệ.
- Client: là những ứng dụng bên thứ 3 muốn truy cập vào phần tài nguyên được chia sẻ với tư cách của người sở hữu (resource owner) và tất nhiên trước khi truy cập ứng dụng cần được sự ủy quyền của user.
- Authorization server: làm nhiệm vụ xác thực, kiểm tra thông tin mà user gửi đến từ đó cấp quyền truy cập cho ứng dụng bằng việc sinh ra các đoạn mã access token. Đôi khi authorization server cũng chính là resource server.

OAuth 2.0 cung cấp nhiều tùy chọn khác nhau về cách có thể triển khai quy trình ủy quyền.

Đặc điểm kỹ thuật có bốn loại tài trợ và IETF đã xuất bản hai hồ sơ bổ sung, một khuôn khổ để sử dụng xác nhận với OAuth 2.0 cũng như soạn thảo thông số kỹ thuật để sử dụng mã xác thực tin nhắn (MAC) với mã thông báo OAuth 2.0. Tuy nhiên, một bức tranh chung về luồng giao thức có thể được trừu tượng hóa.

❖ Luồng bao gồm các bước sau:



Hình 2.2. Luồng giao thức trừu tượng

A: "Khách hàng yêu cầu ủy quyền từ chủ sở hữu tài nguyên. Yêu cầu ủy quyền có thể được gửi trực tiếp cho chủ sở hữu tài nguyên (như được hiển thị), hoặc tốt hơn là gián tiếp thông qua máy chủ ủy quyền làm trung gian."

B: "Khách hàng nhận được một khoản cấp phép, là thông tin xác thực đại diện cho sự ủy quyền của chủ sở hữu nguồn, được thể hiện bằng cách sử dụng một trong bốn loại tài trợ được xác định trong điều này đặc điểm kỹ thuật hoặc sử dụng một loại tài trợ mở rộng. Loại tài trợ ủy quyền phụ thuộc về phương pháp được khách hàng sử dụng để yêu cầu ủy quyền và các loại được hỗ trợ bởi máy chủ ủy quyền."

C: "Máy khách yêu cầu mã thông báo truy cập bằng cách xác thực với máy chủ ủy quyền và xuất trình khoản cấp phép."

D: "Máy chủ ủy quyền xác thực máy khách và xác thực việc cấp phép, và nếu hợp lệ, hãy phát hành một mã thông báo truy cập."

E: "Máy khách yêu cầu tài nguyên được bảo vệ từ máy chủ tài nguyên và xác thực bằng cách xuất trình mã thông báo truy cập"

F: "Máy chủ tài nguyên xác nhận mã thông báo truy cập và nếu hợp lệ, sẽ phục vụ yêu cầu."

❖ Tài nguyên máy chủ khách hàng

Khách hàng phải được đăng ký trong máy chủ tài nguyên mà bạn muốn truy cập. Khách hàng được xác định bởi `client_id` và thường có thêm các URL chuyển hướng để sử dụng trong việc trao đổi mã thông báo và truy cập `client_secret`. Quá trình đăng ký một ứng dụng khách mới không được xác định trong các thông số kỹ thuật của OAuth2. Tài nguyên máy chủ chịu trách nhiệm yêu cầu dữ liệu cần thiết để đăng ký máy khách.

OAuth2 xác định hai loại khách hàng tùy thuộc vào khả năng của họ để duy trì thông tin xác thực của khách hàng an toàn và không bị lộ. Các loại khách hàng là:

- **Bảo mật:** Những khách hàng có thể duy trì thông tin xác thực một cách an toàn và bảo mật. Những khách hàng này là những người được thực hiện trên các máy chủ an toàn.
- **Công chúng:** Có những khách hàng không thể duy trì tính bảo mật của thông tin đăng nhập. Khách hàng là web thuần túy (ứng dụng dựa trên trình duyệt) hoặc các ứng dụng và thiết bị đầu cuối gốc.

❖ Điểm cuối trên nhà cung cấp dịch vụ

Nhà cung cấp có thể xác định hai điểm cuối để cho phép tất cả xác thực luồng ủy quyền (trong một số trường hợp, cả hai đều cần thiết và trong những trường hợp khác, chỉ cần thiết bị đầu cuối mã thông báo). Một trong số họ sẽ chịu trách nhiệm xác thực thông tin và người kia sẽ chịu trách nhiệm phát hành mã thông báo truy cập.

- **Điểm cuối ủy quyền:** Điểm truy cập này phải chịu trách nhiệm xác thực các máy khách và chủ sở hữu tài nguyên. Khi một khách hàng muốn truy cập vào một tài nguyên, đó là lúc nhà cung cấp xác nhận cả khách hàng và chủ sở hữu, nó tạo ra các nhượng bộ để truy cập các tài nguyên được bảo vệ. Thông thường nhận được (ngoài các thông tin xác thực cần thiết) chuyển hướng URL để chuyển hướng luồng khi chủ sở hữu và khách hàng đã được xác thực, khi điều này xảy ra, chuyển hướng url() sẽ nhận được các nhượng bộ (thường ở định dạng mã).
- **Điểm cuối mã thông báo:** Điểm này chịu trách nhiệm cấp mã thông báo truy cập, thường để đổi lấy các ưu đãi do điểm cuối cấp phép phát hành trước đó.

❖ Nhận quyền truy cập vào nguồn cung cấp được bảo vệ

Khi khách hàng muốn truy cập tài nguyên trên chủ sở hữu, điều đầu tiên bạn cần là yêu cầu quyền đối với chủ sở hữu tài nguyên đó. Bước này được gọi là “**cấp phép**”. Đối với một máy chủ tài nguyên cung cấp cho bạn mã thông báo truy cập, điều cần thiết là phải gửi thông tin xác thực, cấp phép cho ứng dụng khách đang cố gắng truy cập dữ liệu. Một nhà cung cấp dịch vụ không nhất thiết phải thực hiện tất cả các loại nhượng bộ (loại tài trợ).

Có bốn loại cấp phép do OAuth2 hỗ trợ và chúng xác định cách lấy mã thông báo truy cập hợp lệ. Mỗi loại ủy quyền được thiết kế để đáp ứng nhu cầu của các loại khách hàng khác nhau.

❖ Quyền trong phạm vi

OAuth2 xác định một tham số có thể tham gia vào việc thương lượng ủy quyền được gọi là phạm vi. Nhà cung cấp chịu trách nhiệm xác định các phạm vi khác nhau và cách xin phép để truy cập từng phạm vi đó. Thông qua một cơ chế xác định phạm vi quyền truy cập được cấp cho khách hàng và chỉ có thể cấp quyền truy cập vào một số tài nguyên hệ thống nhất định.

2.1.4. Các tính năng bảo mật

2.1.4.1. Grant type

Phần đầu tiên của việc triển khai Auth 2.0 liên quan đến việc chọn các loại tài trợ được hỗ trợ bởi việc triển khai. Bốn loại hỗ trợ được xác định bởi thông số kỹ thuật OAuth 2.0 là mã ủy quyền, ngầm định, thông tin xác thực mật khẩu của chủ sở hữu tài nguyên và thông tin xác thực ứng dụng khách. Như đã lưu ý trước đây, cũng có một cơ chế mở rộng giúp bạn có thể xác định các kiểu bổ sung, hai trong số đó đã được IETF chỉ định. Mô hình triển khai trong nghiên cứu này nhằm mục đích dễ thực hiện và dễ hiểu mà không ảnh hưởng đến bảo mật. Càng ít loại tài trợ được triển khai thì mô hình sẽ càng đơn giản và nhà phát triển càng phải đối mặt với ít cân nhắc bảo mật hơn. Do đó, mô hình nhằm mục đích bao gồm ít loại tài trợ nhất có thể để hỗ trợ an toàn cho hầu hết các trường hợp sử dụng. Vì triển khai OAuth 2.0 sẽ là một API ứng dụng web với người dùng cuối, hỗ trợ cho thông tin xác thực mật khẩu của chủ sở hữu tài nguyên và các loại cấp thông tin xác thực ứng dụng khách không được ưu tiên.

Khi sử dụng thông tin đăng nhập mật khẩu của chủ sở hữu tài nguyên để lấy mã thông báo truy cập, ứng dụng sẽ sử dụng thông tin đăng nhập của người dùng cuối (nghĩa là kết hợp mật khẩu và tên người dùng). Do đó, người dùng phải chia sẻ thông tin đăng nhập của họ với ứng dụng khách, điều này đòi hỏi

mức độ tin cậy cao giữa chủ sở hữu tài nguyên và ứng dụng khách. Đây không phải là luồng mong muốn, khi API có thể truy cập được đối với các nhà phát triển ứng dụng khách độc lập và nên được sử dụng chỉ với các ứng dụng máy khách gốc do nhà cung cấp API phát triển. Loại cấp thông tin xác thực máy khách có thể được sử dụng khi tài nguyên được bảo vệ thuộc sở hữu của máy khách. Nó thường thích hợp để xác thực dựa trên máy tính. Tuy nhiên, trong trường hợp của chúng tôi, có những người dùng cuối tham gia với tư cách là chủ sở hữu tài nguyên.

Do đó, các loại tài trợ nổi bật cần xem xét cho một API dịch vụ web với người dùng cuối là mã xác thực và ngầm định. Trong số hai điều này, việc cấp mã xác thực có ít bảo mật hơn cần nhắc và hỗ trợ nhiều trường hợp sử dụng hơn so với loại tài trợ ngầm định. Loại tài trợ ngầm chỉ hỗ trợ mã thông báo truy cập và không hỗ trợ mã thông báo làm mới. Loại cấp mã xác thực hỗ trợ cả hai. Loại tài trợ ngầm được tối ưu hóa cho các khách hàng công khai, những khách hàng không có khả năng duy trì tính bảo mật của các mã thông báo truy cập. Những ứng dụng khách này thường được chạy trực tiếp trong trình duyệt web bằng JavaScript. Mặt khác, loại cấp mã ủy quyền được tối ưu hóa cho các khách hàng bí mật, có thể duy trì tính bảo mật của việc truy cập và làm mới mã thông báo.

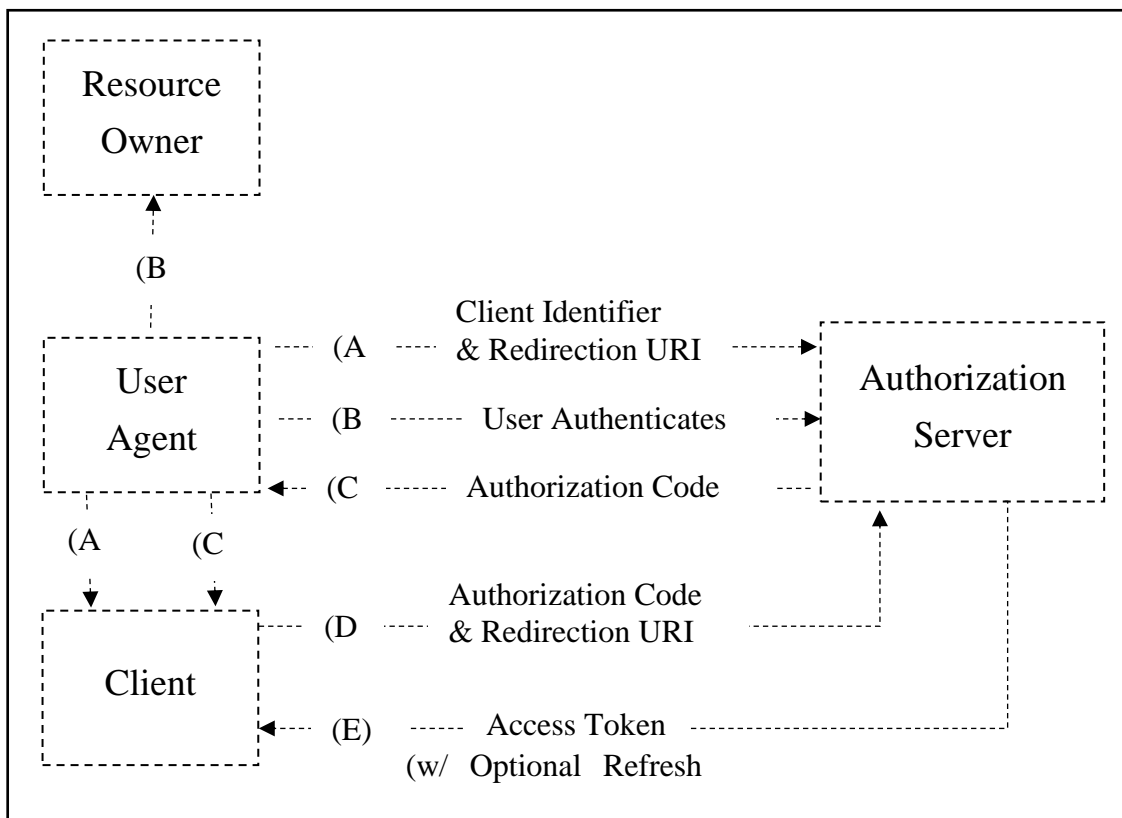
Trong mã thông báo truy cập loại cấp phép ngầm được mã hóa thành một URI chuyển hướng mà máy chủ ủy quyền trả lại cho máy khách trong phần cuối của quy trình ủy quyền. Do đó, nó có thể được tiếp xúc với chủ sở hữu tài nguyên và các ứng dụng khác trên cùng một thiết bị, ví dụ: mặc dù cuộc tấn công Cross-site scripting (XSS). Khi sử dụng loại cấp mã ủy quyền, mã thông báo truy cập có thể được truyền trực tiếp đến máy khách. Điều này giúp loại bỏ nguy cơ lộ mã truy cập khi chuyển nó qua tác nhân người dùng của chủ sở hữu tài nguyên. Loại cấp mã xác thực cũng giúp xác thực máy khách.

Do đó, mô hình triển khai sẽ chỉ bao gồm một loại cấp: mã ủy quyền. Việc chỉ sử dụng một loại tài trợ sẽ đơn giản hóa quá trình thực hiện. Nó cũng loại bỏ nhu cầu đăng ký luồng ủy quyền của khách hàng. Sau đó, mô hình triển khai sẽ chỉ bao gồm hai luồng: Luồng mã ủy quyền và quy trình trao đổi mã thông báo làm mới sang mã thông báo truy cập mới.

2.1.4.2. Authorization Code

Loại truy cập này được sử dụng khi cố gắng truy cập tài nguyên máy chủ ứng dụng web. Chúng tôi phải luôn ghi nhớ rằng chủ sở hữu tài nguyên xác thực

và xác thực máy chủ ủy quyền, thông tin đăng nhập không bao giờ được tiết lộ cho khách hàng. Sau khi chủ sở hữu của các tài nguyên đã được xác thực bởi nhà cung cấp, bằng URL chuyển hướng sẽ trả lại cho khách hàng một mã truy cập có thể được trao đổi lấy mã truy cập.



Hình 2.3. Luồng mã ủy quyền

- A: Khách hàng khởi tạo luồng hướng tác nhân người dùng (trình duyệt) đến điểm cuối ủy quyền của chủ sở hữu tài nguyên. Máy khách bao gồm client_id, phạm vi ủy quyền và url() mà máy chủ ủy quyền sẽ chuyển hướng đến cuối yêu cầu.
- B: Máy chủ ủy quyền xác định chủ sở hữu của tài nguyên bằng cách sử dụng trình duyệt và thiết lập xem chủ sở hữu có cho phép hay không khách hàng.
- C: Giả sử rằng chủ sở hữu đã cấp quyền, máy chủ ủy quyền sẽ chuyển hướng chủ sở hữu tác nhân người dùng đến redirect_uri được gửi ở bước A. Trong chuyển hướng này bao gồm một mã ủy quyền hợp lệ để trao đổi cho ủy quyền mã thông báo truy cập.
- D: Khách hàng đưa ra yêu cầu tại điểm cuối mã thông báo, thông báo bằng chứng xác thực và mã ủy quyền của họ.
- E: Máy chủ ủy quyền xác minh thông tin đăng nhập và mã ủy quyền và nếu thành công sẽ trả về mã thông báo truy cập hợp lệ ...

2.1.4.3. Yêu cầu mã hóa bằng Message Authentication Code (MAC)

Tính bảo mật của việc triển khai có thể được cải thiện hơn nữa bằng cách sử dụng "mã thông báo bằng chứng" đã ký thay vì mã thông báo mang văn bản thuần túy. Việc triển khai có thể được thực hiện, ví dụ: bằng cách sử dụng Mã xác thực tin nhắn (MAC). Điều này sẽ giảm thiểu một phạm vi lớn của các mối đe dọa và được Hammer ủng hộ trong số những người khác. Tuy nhiên, kiểu cấp mã thông báo MAC OAuth 2.0 vẫn đang được soạn thảo và việc triển khai mã thông báo MAC làm tăng thêm mức độ phức tạp cho việc triển khai, điều này không hoàn toàn cần thiết cho việc bảo mật của việc triển khai.

Thay vì mã hóa thông tin mã thông báo truy cập và chuẩn bị cho việc đánh cắp mã thông báo truy cập, việc triển khai hiện tại sẽ nhằm mục đích bảo vệ tất cả các thông tin liên lạc khỏi bị nghe trộm và đánh cắp mã thông báo. Điều này được hỗ trợ bởi việc sử dụng loại cấp mã xác thực và sử dụng TLS nghiêm ngặt trong việc bảo vệ lưu lượng.

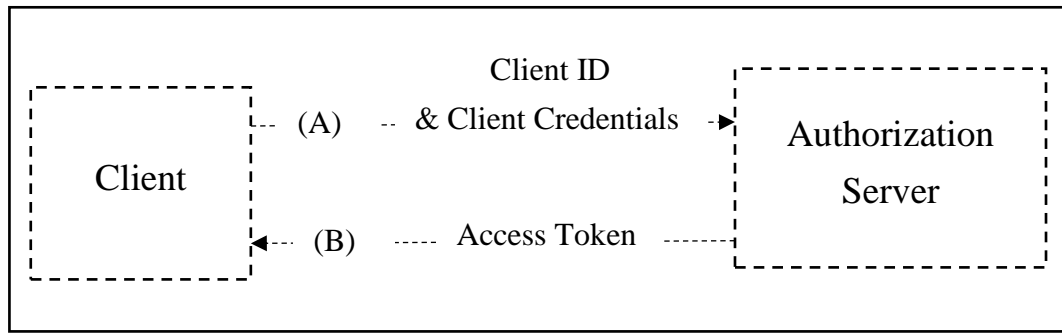
2.1.4.4. Implicit

Loại cấp quyền ngầm được sử dụng để nhận mã thông báo truy cập (loại này không hỗ trợ phát hành mã thông báo làm mới) và được tối ưu hóa cho công chúng khách hàng được biết để vận hành một URI chuyển hướng cụ thể. Những khách hàng này thường được thực hiện trong trình duyệt sử dụng ngôn ngữ kịch bản chẳng hạn như JavaScript.

Vì đây là luồng dựa trên chuyển hướng, nên máy khách phải có khả năng tương tác với tác nhân người dùng của chủ sở hữu tài nguyên (thường là web trình duyệt) và có khả năng nhận các yêu cầu đến (thông qua chuyển hướng) từ máy chủ ủy quyền.

Không giống như loại cấp mã ủy quyền, trong đó khách hàng thực hiện yêu cầu ủy quyền riêng biệt và mã thông báo truy cập, khách hàng nhận được mã thông báo truy cập là kết quả của ủy quyền yêu cầu.

Loại cấp quyền ngầm không bao gồm xác thực ứng dụng khách và phụ thuộc vào sự hiện diện của chủ sở hữu tài nguyên và đăng ký URI chuyển hướng. Bởi vì mã thông báo truy cập được mã hóa vào URI chuyển hướng, nó có thể được hiển thị cho chủ sở hữu tài nguyên và khác các ứng dụng nằm trên cùng một thiết bị.



Hình 2.4. Luồng Implicit

- A: Máy khách yêu cầu máy chủ ủy quyền một mã thông báo truy cập.
- B: Máy chủ ủy quyền xác thực thông tin xác thực của máy khách, do đó trả về mã thông báo truy cập hợp lệ trực tiếp.

2.1.4.5. Bảo mật lưu lượng bằng Transport Layer Security (TLS)

Không thể quá nhấn mạnh sự cần thiết của việc sử dụng TLS với OAuth 2.0. Việc triển khai TLS yêu cầu thêm tài nguyên và kiến thức vì nhà phát triển phải có được chứng chỉ TLS và đã định cấu hình chứng chỉ này trên máy chủ. Ngoài ra, hỗ trợ TLS và kiểm tra chứng chỉ sẽ được triển khai cho máy chủ và các ứng dụng khách. Trong tình huống này, nhà phát triển có thể bị cám dỗ không sử dụng TLS và sử dụng giao thức HTTP không được bảo vệ hoặc bỏ qua một số tính năng TLS như kiểm tra tính xác thực của chứng chỉ.

Tuy nhiên, như rõ ràng từ đặc điểm kỹ thuật và được xác minh thêm bởi Xu, Niu và Meng OAuth 2.0 không có cơ chế tích hợp nào để bảo vệ lưu lượng giữa máy khách và máy chủ. Do đó, không có TLS, tất cả thông tin ủy quyền sẽ được gửi rõ ràng và có sẵn cho bất kỳ ai nghe trộm lưu lượng truy cập. Chỉ cần tuân theo đặc điểm kỹ thuật và sử dụng TLS, một số vấn đề bảo mật có thể tránh được.

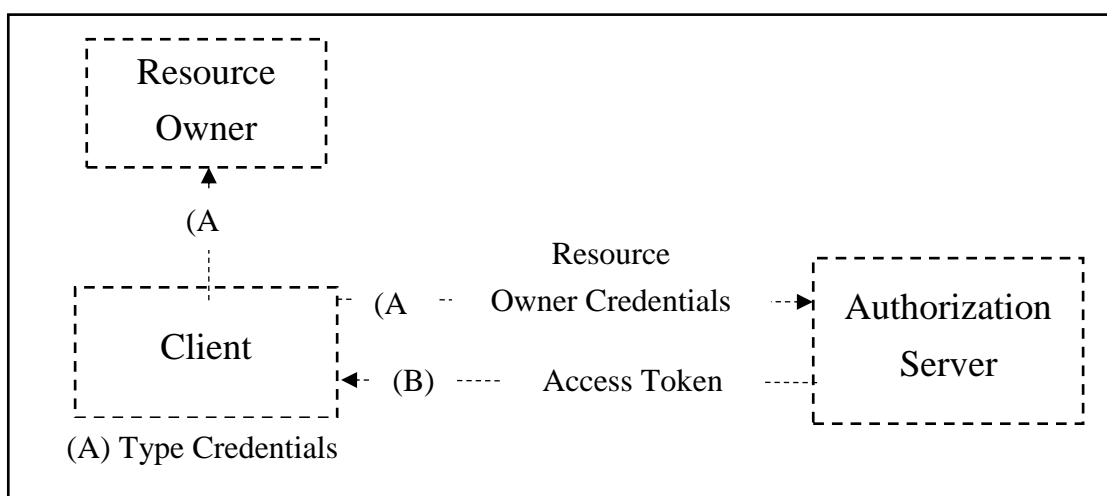
Theo đặc điểm kỹ thuật, máy chủ ủy quyền phải yêu cầu TLS tại điểm cuối cấp quyền và mã thông báo vì chúng liên quan đến việc chuyển quyền truy cập và làm mới mã thông báo. Tuy nhiên, nó không được yêu cầu tại điểm cuối chuyển hướng, vì nó sẽ gây khó khăn cho nhiều nhà phát triển khách hàng. Thay vào đó, các rủi ro liên quan đến khả năng nghe trộm mã xác thực được giảm thiểu bằng cách mã ủy quyền hết hạn rất ngắn và cơ chế thu hồi tất cả các mã thông báo được cấp cùng với mã ủy quyền, nếu nó được sử dụng nhiều lần.

Hơn nữa, máy khách phải xác thực chuỗi chứng chỉ TLS của máy chủ ủy quyền để ngăn chặn các cuộc tấn công kẻ trung gian bằng cách sử dụng chiếm quyền điều khiển DNS. Nếu không, kẻ tấn công có thể giành quyền truy cập vào dữ liệu được chuyển bao gồm mã ủy quyền và mã thông báo.

2.1.4.6. Resource Owner Credentials

Loại cấp quyền thông tin mật khẩu của chủ sở hữu tài nguyên là phù hợp trong trường hợp chủ sở hữu tài nguyên có mối quan hệ tin cậy với khách hàng, chẳng hạn như hệ điều hành thiết bị hoặc một ứng dụng đặc quyền cao. Máy chủ ủy quyền cần đặc biệt cẩn thận khi cho phép loại cấp này và chỉ cho phép nó khi các luồng khác không khả thi.

- Loại tài trợ này phù hợp cho khách hàng có khả năng có được thông tin đăng nhập của chủ sở hữu tài nguyên (tên người dùng và mật khẩu, thường sử dụng một hình thức tương tác). Nó cũng được sử dụng để di chuyển các khách hàng hiện tại sử dụng các lược đồ xác thực trực tiếp như HTTP Basic hoặc Digest xác thực thành OAuth bằng cách chuyển đổi thông tin đăng nhập được lưu trữ thành truy cập thể.



Hình 2.5. Luồng Resource Owner Credentials

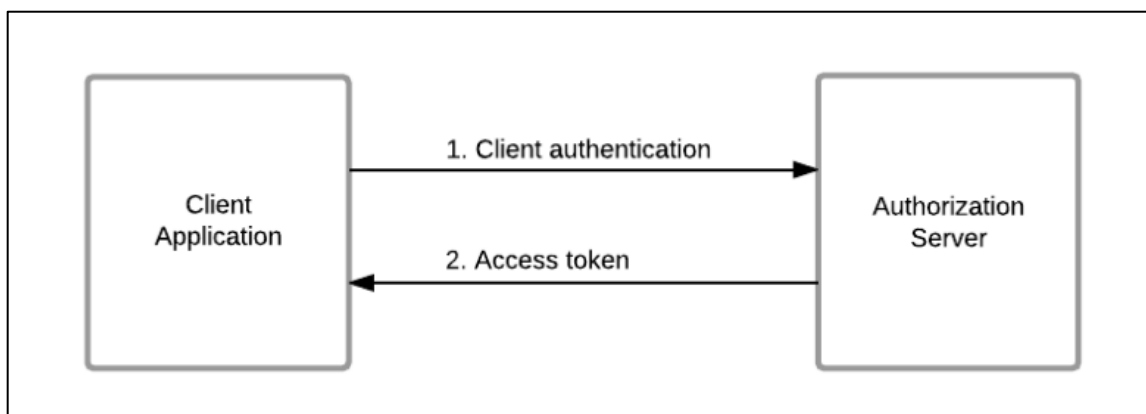
- A: Máy khách yêu cầu máy chủ ủy quyền một mã thông báo truy cập bằng cách gửi trực tiếp chủ sở hữu tài nguyên thông tin xác thực của người dùng.
- B: Máy chủ ủy quyền sau đó xác thực thông tin xác thực của khách hàng, trả về mã thông báo truy cập hợp lệ trực tiếp.

2.1.4.7. Client Authentication

Để có thể xác thực khách hàng, tất cả khách hàng phải được cấp bằng chứng xác thực khách hàng bao gồm ID khách hàng và bí mật khách hàng. Chúng phải được chuyển tới điểm cuối mã thông báo trong phần thân yêu cầu bằng cách sử dụng các tham số `client_id` và `client_secret`.

Trái ngược với khuyến nghị của đặc tả OAuth 2.0, việc triển khai sẽ không hỗ trợ lược đồ xác thực HTTP cơ bản. Nó không cung cấp bất kỳ bảo mật bổ sung nào cho việc triển khai. "Lược đồ xác thực cơ bản không phải là một phương pháp xác thực người dùng an toàn. Nó dẫn đến việc truyền tải về cơ bản rõ ràng mật khẩu của người dùng qua mạng vật lý."

Do cách thức hoạt động của các trình duyệt web hiện đại, việc hỗ trợ xác thực "HTTP Basic" thậm chí có thể cản trở sự bảo mật của máy khách. Điều này đúng đặc biệt nếu việc triển khai sau đó được mở rộng để hỗ trợ loại tài trợ ngầm. Nhiều trình duyệt web tự động lưu thông tin xác thực cơ bản vào "bộ nhớ cache" trong suốt thời gian của phiên người dùng. Họ cũng thường cung cấp cho người dùng khả năng lưu các thông tin đăng nhập này. Điều này có nghĩa là thông tin xác thực có thể được sử dụng lại một cách âm thầm với bất kỳ yêu cầu nào khác đến máy chủ, khiến cho các cuộc tấn công giả mạo yêu cầu chéo (CSRF) có thể xảy ra. Ngoài ra, thông tin xác thực có thể bị đánh cắp bởi người dùng hoặc quy trình khác.



Hình 2.6. Luồng hoạt động của Client Authentication

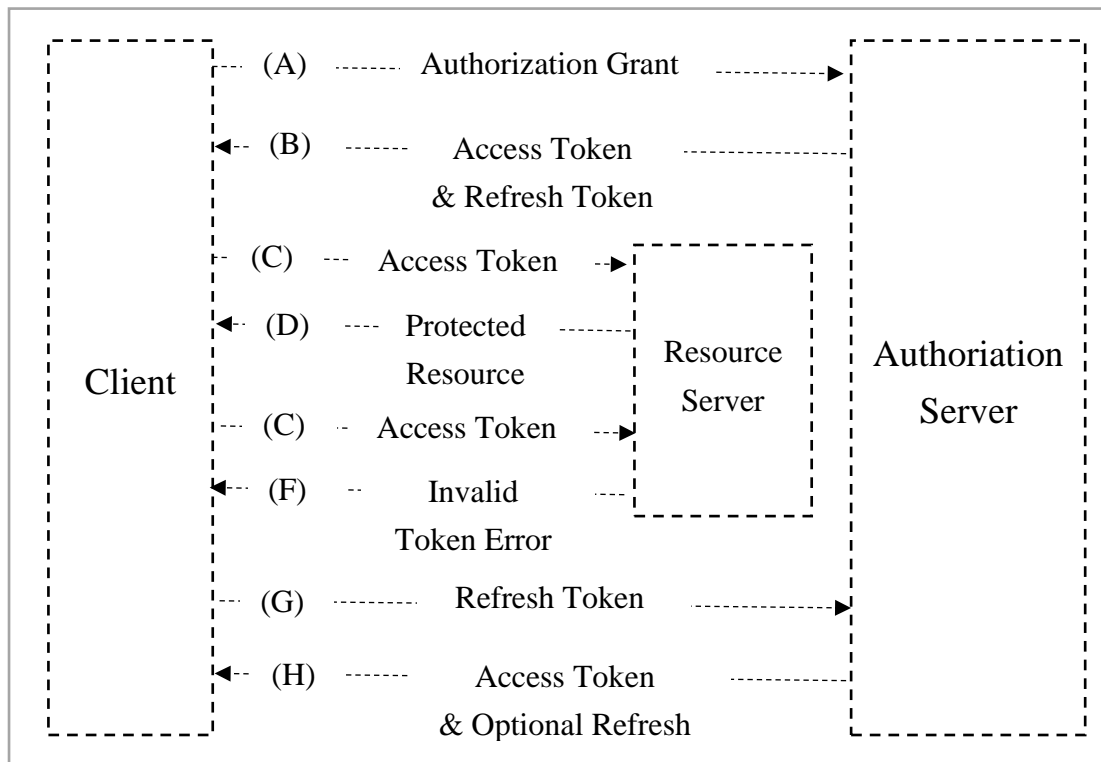
Luồng hoạt động của "Client Credentials" như sau:

- Máy khách xác thực với máy chủ ủy quyền và yêu cầu mã thông báo truy cập từ điểm cuối mã thông báo.
- Máy chủ ủy quyền xác thực ứng dụng khách và nếu hợp lệ, phát hành mã thông báo truy cập

2.1.4.8. Refresh Token

Mã thông báo truy cập phải có ngày hết hạn sau khi chúng được coi là đã hết hạn, nhà cung cấp phải từ chối yêu cầu này, buộc khách hàng phải lấy mã thông báo truy cập mới. Để tránh thực hiện lại tất cả quy trình xác thực cần thiết để lấy mã thông báo, thật tiện lợi khi sử dụng cơ chế làm mới mã thông báo. Khi máy chủ ủy quyền cung cấp mã thông báo truy cập, bạn có thể bao gồm mã làm mới được liên kết với mã thông báo truy cập.

Với cơ chế mã thông báo làm mới, có thể tạo mã thông báo truy cập mới bằng cách sử dụng mã làm mới được tạo trong xác thực ứng dụng khách trước đó do chủ sở hữu tài nguyên thực hiện.



Hình 2.7. Làm mới mã thông báo truy cập đã hết hạn

- A: Khách hàng yêu cầu một mã thông báo truy cập đến nhà cung cấp, trình bày một khoản cấp phép.
- B: Máy chủ ủy quyền xác thực yêu cầu và trả về mã thông báo truy cập.
- C: Máy khách yêu cầu tài nguyên đối với tài nguyên máy chủ trình bày mã thông báo truy cập.
- D: Máy chủ tài nguyên xác nhận mã thông báo truy cập và nếu hợp lệ sẽ trả về các tài nguyên được bảo vệ.
- E: Các bước C và D được lặp lại cho đến khi máy chủ tài nguyên phát hiện ra quyền truy cập mã thông báo đã hết hạn hoặc không hợp lệ.

lệ. Khi phản hồi của khách hàng, mã thông báo truy cập không hợp lệ sẽ chuyển sang bước G.

- F: Khi mã thông báo truy cập không hợp lệ, máy chủ tài nguyên sẽ thông báo cho máy khách.
- G: Khách hàng đưa ra yêu cầu lấy mã thông báo truy cập mới, trình bày mã thông báo làm mới.
- H: Máy chủ xác thực ủy quyền máy khách và xác thực mã thông báo làm mới. Nếu mọi thứ đều hợp lệ, hãy trả về mã thông báo truy cập và tùy chọn là mã thông báo làm mới.

2.2. Xác thực Single Sign On (SSO)

2.2.1. Thuật ngữ SSO

❖ Định nghĩa 1

- **IdP** là viết tắt của Identity Provider. Nhà cung cấp danh tính (IdP) chịu trách nhiệm về quá trình xác thực và xử lý việc lưu trữ thông tin người dùng dưới dạng thuộc tính trong mã thông báo nhận dạng. Khi người dùng xác thực, IdP sẽ tạo một đối tượng chứa thông tin của người dùng, đối tượng này có thể được sử dụng khi nhà cung cấp dịch vụ yêu cầu thuộc tính người dùng.
- IdP có thể được so sánh với chính phủ. Họ có dữ liệu cá nhân nhất định về công dân của họ. Khi bạn muốn đi du lịch đến một quốc gia khác, bạn cần phải có hộ chiếu. Sử dụng chính phủ, bạn có thể nhận được một hộ chiếu, chứng minh danh tính của bạn.

❖ Định nghĩa 2

- **AS** là viết tắt của Authorization Server. Máy chủ ủy quyền cấp mã thông báo truy cập cho máy khách sau khi xác thực thành công. Khi IdP trả lại mã thông báo truy cập, thì IdP cũng hoạt động như một máy chủ ủy quyền.
- **AS** có thể được so sánh với một công ty du lịch cung cấp vé máy bay cho khách hàng. Vé này chứng minh rằng bạn có quyền truy cập vào một chuyến bay nhất định.

❖ Định nghĩa 3

- **SP** là viết tắt của Service Provider. Nhà cung cấp dịch vụ cung cấp một dịch vụ nhất định và nói chung được bảo vệ bởi một loại nhân

viên bảo vệ. Khi người dùng muốn sử dụng dịch vụ, người bảo vệ sẽ yêu cầu thông tin nhận dạng từ người dùng.

- **SP** có thể được so sánh với một sân bay. Trước khi có thể bắt máy bay, bạn phải chứng minh với bảo vệ rằng bạn nên có mặt trên chuyến bay đó bằng cách sử dụng vé máy bay và hộ chiếu của bạn để chứng minh rằng vé là của bạn. Dựa trên hộ chiếu và vé của bạn, người bảo vệ sẽ quyết định bạn có thể vào chuyến bay hay không.

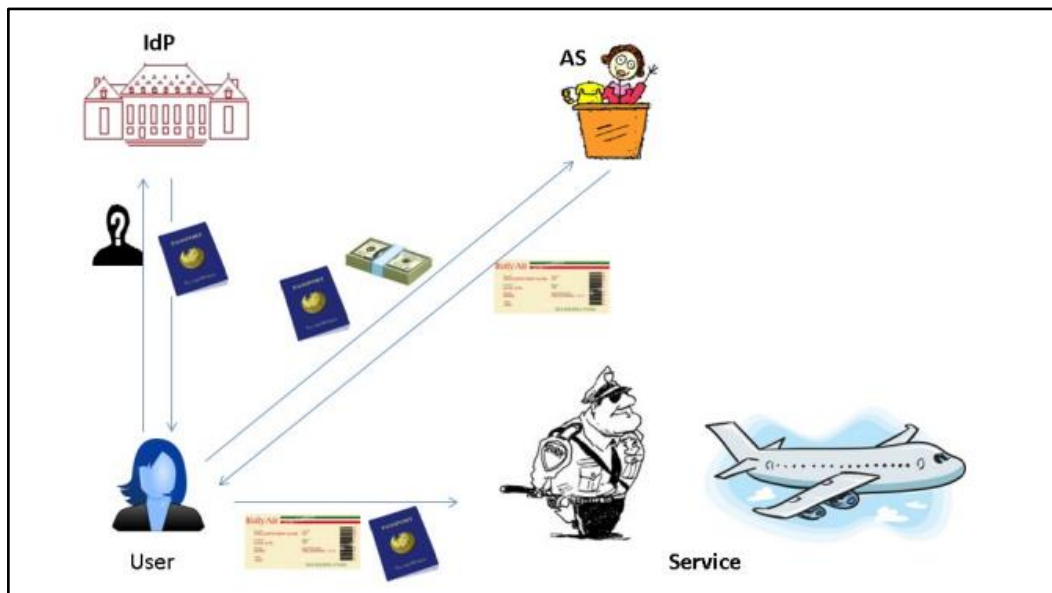
❖ Định nghĩa 4

- Khi chúng ta nói về một **Client**, chúng ta không có nghĩa là một người hoặc một công ty mà là thiết bị (máy tính / máy tính xách tay / máy tính bảng / điện thoại di động / v.v.) thông qua đó người dùng truy cập vào một dịch vụ.

❖ Định nghĩa 5

User agent là phần mềm trên máy khách, được sử dụng để giao tiếp với các máy chủ trong giao thức. Vì chúng tôi đang tìm kiếm một giải pháp kết nối các dịch vụ web với nhau nên tác nhân người dùng thường là một trình duyệt web.

Dưới đây là một ví dụ đơn giản về việc triển khai SSO. Hình vẽ minh họa một người dùng muốn thực hiện chuyến bay đến một điểm đến nhiệt đới. Để có được quyền truy cập vào chuyến bay, dịch vụ nhà cung cấp muốn biết ai đang lên chuyến bay và liệu người này có được ủy quyền hay không làm như vậy. SP yêu cầu hộ chiếu và vé máy bay. Để lấy hộ chiếu hành khách đi đến tòa thị chính (IdP) và yêu cầu một hộ chiếu. Một nhân viên xác minh danh tính và cung cấp một hộ chiếu. Người dùng đến đại lý chuyến bay (AS) và mua vé máy bay. Khi nào người dùng đến sân bay giao vé và hộ chiếu của mình cho nhà cung cấp dịch vụ, ai sẽ cấp hoặc từ chối quyền lên máy bay.



Hình 2.8. Các tác nhân SSO

2.2.2. Định nghĩa SSO

Single Sign-On (SSO), đúng như tên gọi, là cơ chế cho phép người dùng có thể truy cập nhiều trang web, ứng dụng mà chỉ cần đăng nhập một lần. Một khi đã được định danh ở một trang website A, thì cũng sẽ được định danh tương tự ở website B mà không cần lặp lại thao tác đăng nhập.

Hiện tại có nhiều cách và nhiều ngôn ngữ lập trình có thể triển khai SSO:

❖ Central Authentication Service (CAS)

CAS là một giải pháp SSO mã nguồn mở được phát triển bởi đại học Yale, với các tính năng như sau:

- CAS hỗ trợ nhiều thư viện phía máy khách được viết bởi nhiều ngôn ngữ: PHP, Java, PL/SQL. Nó lấy thông tin SSO thông qua cookie. Cookie này sẽ bị hủy khi người dùng đăng xuất khỏi CAS hoặc đóng trình duyệt. Cookie được sinh ra bởi CAS, còn được gọi là Ticket Granting Cookie (TGC) chứa một ID duy nhất.
- CAS cung cấp nhiều trình quản lý xác thực (authenticate handler) khác nhau. CAS xác thực nhiều loại thông tin người dùng như tên truy cập/mật khẩu, chứng chỉ khóa công khai X509,... để xác thực những thông tin người dùng khác nhau này, CAS sử dụng những trình quản lý xác thực tương ứng.
- CAS cung cấp tính năng “Remember me”. Người phát triển có thể cấu hình tính năng này trong nhiều file cấu hình khác nhau và khi người dùng chọn “Remember me” trên khung đăng nhập, thì thông tin đăng

nhập sẽ được ghi nhớ với thời gian được cấu hình. Khi người dùng mở trình duyệt thì CAS sẽ chuyển đến service URL tương ứng mà không cần hiển thị khung đăng nhập.

Nguyên tắc chứng thực người dùng với máy chủ CAS hoạt động như sau:

- Người dùng nhập tên truy cập/mật khẩu vào khung đăng nhập. Các thông tin này được truyền cho CAS máy chủ thông qua giao thức HTTPS.
- 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 đăng nhập với tất cả các ứng dụng

2.2.3. Cơ chế hoạt động

2.2.3.1. Đặc điểm Cookie và Session

❖ Đặc điểm của Cookie

- Cookie được lưu trữ trên trình duyệt, khi người dùng sử dụng trình duyệt truy cập vào một website nào đó thì server sẽ gửi cookie về trình duyệt và lưu trữ trực tiếp trên máy người dùng (client). Cookie sẽ khác nhau cho mỗi loại trình duyệt, IP người dùng cũng như server của website. Cookie được tạo ra bởi website và gửi tới browser, do vậy 2 website khác nhau (cho dù cùng host trên 1 server) sẽ có 2 cookie khác nhau gửi tới trình duyệt. Mỗi trình duyệt sẽ có cách quản lý và lưu trữ cookie theo cách riêng của mình, cho nên 2 trình duyệt cùng truy cập vào 1 website sẽ nhận được 2 cookie khác nhau. Mỗi lần có request lên website, trình duyệt sẽ gửi cookie lên server, server sẽ có cơ chế “đọc” cookie và xử lý cho tác vụ nào đó.
- Mỗi cookie được website gửi đến trình duyệt thường có các thành phần chính sau:
 - Địa chỉ URL của website mà trình duyệt nhận cookie
 - Thời gian tồn tại của Cookie

❖ Đặc điểm của Session

- Session dùng để lưu phiên làm việc của người dùng trên trang website. Khác với Cookie, Session được lưu trữ trên webServer. Session được lưu với một chuỗi ký tự dài gọi là ID, Session kết thúc khi bị xóa hoặc hết phiên làm việc (đóng trình duyệt – Cookie lưu trên file ở Client nên khi đóng trình duyệt sẽ không mất Cookie).

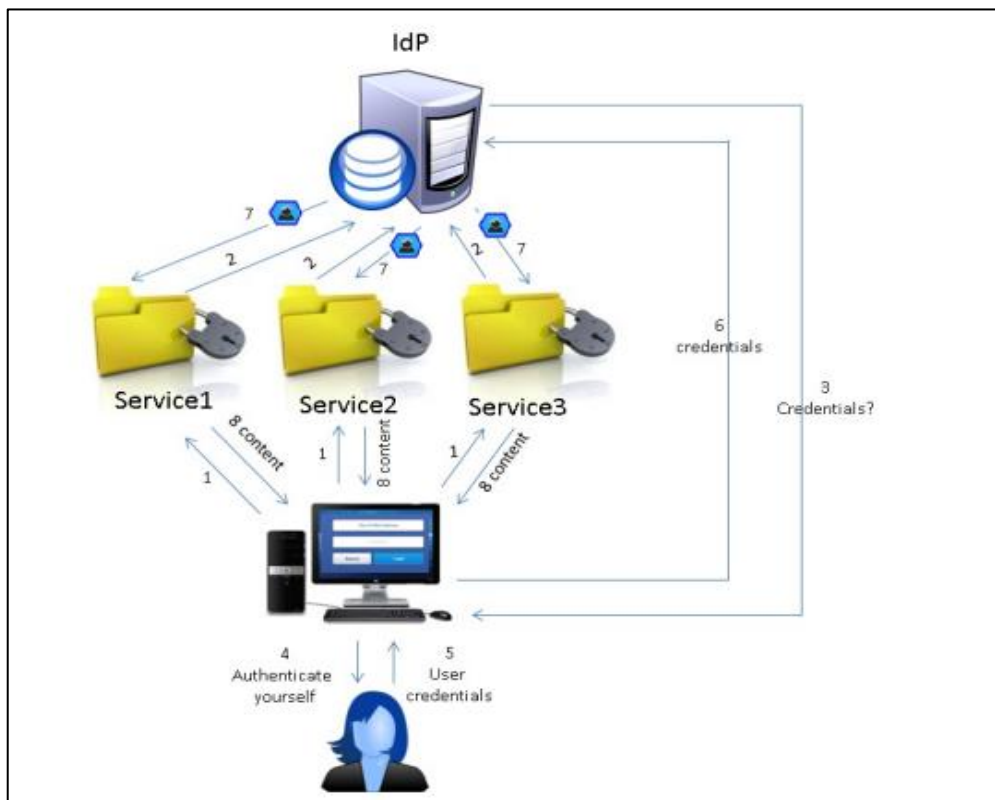
- Sử dụng Captcha cho Login hoặc các Form cần thiết
- Lưu thông tin Client và thực hiện chặn các địa chỉ IP ở những nơi hay tấn công vào hệ thống
- Thực hiện lọc dữ liệu đầu vào, đầu ra, validate dữ liệu...
- Khi có lỗi chỉ trả về những thông tin cho phép hiển thị, không nên hiển thị đầy đủ thông tin lỗi cho người dùng
- Triển khai trên môi trường có độ an toàn cao ví dụ như Linux, Ubuntu...

2.2.3.2. Hệ thống nhận dạng liên kết

Hệ thống nhận dạng liên kết (Federated Identity Glossary) là nơi tập trung và liên kết thông tin người dùng. Có 4 yếu tố nền tảng cấu thành nên hệ thống này:

- Xác thực (Authentication): kiểm tra thông tin đăng nhập và tiến hành định danh người dùng.
- Phân quyền (Authorization): dựa trên thông tin định danh để kiểm tra quyền truy cập của user.
- Trao đổi thông tin người dùng (User attributes exchange): Mỗi hệ thống con sẽ cần và lưu trữ các thông tin khác nhau của người dùng, tuy nhiên sẽ có các thông tin bị lặp lại, ví dụ như tên, họ.... Do đó, cần có một nơi để tổng hợp lại các thông tin này, và trao đổi cho các hệ thống con.
- Quản lý người dùng (User management): admin có thể quản lý người dùng bằng các thao tác thêm, sửa, xóa... ở các hệ thống con.

Single sign-on là một phần của hệ thống nhận dạng liên kết, có liên quan chặt chẽ với việc xác thực thông tin người dùng. Nó sẽ định danh người dùng, và sau đó chia sẻ thông tin định danh được với các hệ thống con.

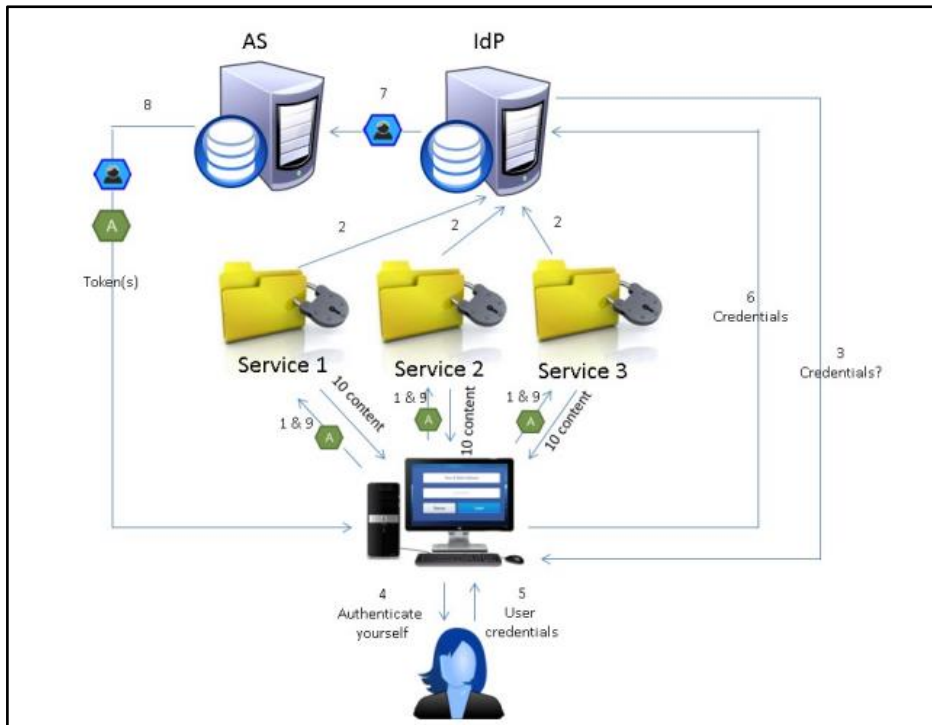


Hình 2.9. Hệ thống nhận dạng liên kết

Các bước cho Hệ thống nhận dạng liên kết gần giống như đối với SSO, tuy nhiên, nó không xử lý ủy quyền (tùy chọn trong SSO) và yêu cầu xác thực lại cho từng dịch vụ. Khi người dùng muốn truy cập vào dịch vụ thứ hai thì quá trình xác thực sẽ bắt đầu lại. Không có bước từ 1 đến 8 vì không có mã thông báo nào có thể được chia sẻ giữa các dịch vụ.

Hệ thống nhận dạng liên kết có thể được coi là người dùng nhận được chìa khóa từ người quản lý để có quyền truy cập vào một phòng cụ thể trong một tòa nhà. Đối với mỗi phòng người dùng muốn vào, anh ta phải đến gặp người quản lý để yêu cầu chìa khóa. Sự khác biệt với quản lý danh tính không liên kết "cơ bản" là: trong giải pháp này, chỉ có một người nói thay vì nhiều người nắm giữ chìa khóa, tất cả đều có một chìa khóa cho một cửa cụ thể.

2.2.3.3. Cơ chế



Hình 2.10. Xác thực SSO

❖ SSO có các bước sau:

- **Bước 1:** Máy khách cố gắng truy cập một dịch vụ. Nếu khách hàng đã có mã thông báo truy cập cho việc này dịch vụ, sau đó mã thông báo được thêm vào yêu cầu. Sau đó, chuyển sang bước 10.
- **Bước 2:** Dịch vụ gọi IdP để xử lý xác thực.
- **Bước 3:** IdP yêu cầu khách hàng cung cấp thông tin đăng nhập.
- **Bước 4:** Máy khách yêu cầu người dùng cung cấp thông tin đăng nhập.
- **Bước 5:** Người dùng giao thông tin đăng nhập.
- **Bước 6:** Khách hàng gửi các thông tin đăng nhập này đến IdP xác nhận các thông tin xác thực.
- **Bước 7:** Nếu thông tin đăng nhập chính xác, một mã thông báo ID sẽ được gửi đến AS; nếu không nó quay trở lại bước 3.
- **Bước 8:** AS thu thập các quyền được gán cho người dùng và tạo mã thông báo truy cập. Các mã thông báo truy cập và mã thông báo ID được gửi đến máy khách.
- **Bước 9:** Máy khách cố gắng truy cập một dịch vụ bằng mã thông báo truy cập.
- **Bước 10:** Dịch vụ cấp quyền truy cập vào dịch vụ.

Quá trình này có thể được coi là nhận được chìa khóa từ người quản lý để có quyền truy cập vào tất cả các phòng trong tòa nhà. Lần đầu tiên bạn muốn vào một phòng, bạn đến gặp người quản lý để yêu cầu một chìa khóa. Sau khi có chìa khóa, bạn có thể vào từng phòng trong tòa nhà bằng chìa khóa duy nhất này.

2.2.4. Ưu và nhược điểm của SSO

– Ưu điểm:

- Lợi thế cho người dùng rõ ràng là tiết kiệm thời gian: Họ chỉ cần nhập mật khẩu một lần và sau đó có thể sử dụng tất cả các dịch vụ khác ngay lập tức.
- Đối với công ty IT, những lợi thế bao gồm cải thiện an toàn và giảm áp lực cho quản trị thấp. Chỉ giao tiếp dữ liệu đăng nhập một lần và cơ sở dữ liệu xác thực tập trung góp phần quan trọng vào việc cải thiện bảo mật. Công việc quản trị cũng đơn giản hơn hơn đáng kể nếu dữ liệu đăng nhập của người dùng cần được thiết lập lại. Khi đăng nhập một lần được sử dụng, dữ liệu người dùng chỉ cần được đặt lại một lần tập trung và không phải cho từng cơ sở dữ liệu riêng lẻ nơi chúng được lưu trữ.
- Trước khi có đăng nhập một lần, một người sử dụng đã phải nhập các tài khoản và mật khẩu cho từng ứng dụng mỗi khi họ đăng nhập vào các ứng dụng khác nhau hoặc các hệ thống trong cùng một phiên. Điều này rõ ràng có thể tốn nhiều thời gian, đặc biệt là trong môi trường doanh nghiệp, nơi mà thời gian là tiền bạc nhưng thời gian là lãng phí bởi vì nhân viên phải đăng nhập mỗi khi họ truy cập vào một hệ thống mới từ máy tính của họ. Đăng nhập một lần thường được thực hiện thông qua một Module xác thực phần mềm riêng biệt hoạt động như một đầu vào cho tất cả các ứng dụng yêu cầu đăng nhập. Các Module xác thực người sử dụng và sau quản lý truy cập vào các ứng dụng khác. Nó hoạt động như một kho dữ liệu chung cho tất cả các thông tin đăng nhập được yêu cầu. Ví dụ: Một Module đăng nhập một lần là hệ thống của Google khi mà người dùng chỉ cần đăng nhập 1 lần thì họ có thể sử dụng các dịch vụ của Google hay Yahoo mà không đòi hỏi đăng nhập 1 lần nữa như Gmail, Google Plus, Youtube..... Trong khi SSO là rất tiện lợi

nhưng cũng nguy hiểm về vấn đề xác thực và bảo mật. Nếu hệ thống đăng nhập một lần bị tấn công, một kẻ tấn công có quyền truy cập không giới hạn cho tất cả các ứng dụng chứng thực của các Module SSO thường là một dự án lớn cần lập kế hoạch cẩn thận trước khi thực hiện.

- Người sử dụng chỉ cần đăng nhập 1 lần để sử dụng nhiều ứng dụng, cải thiện sự bảo mật, tiết kiệm... Dễ dàng kiểm soát, nâng cấp, bảo trì và Thân thiện với người sử dụng.

– Nhược điểm:

- Tính khả dụng của dịch vụ cũng phụ thuộc vào tính khả dụng của hệ thống đăng nhập một lần.
- Trộm cắp khóa truy cập cho phép kẻ tấn công truy cập vào tất cả các hệ thống.
- Bảo mật: Nếu một kẻ xâm nhập làm tổn hại tài khoản của người dùng hoặc mật khẩu, kẻ xâm nhập có thể có rộng rãi và dễ dàng truy cập vào rất nhiều ứng dụng.
- Chi phí: triển khai SSO có thể tốn kém, cả về chi phí để mua và nguồn nhân lực để triển khai. Hai yếu tố SSO là tốt nhất, nơi truy cập được cấp dựa trên sự kết hợp đối với những gì người sử dụng biết (mật khẩu hoặc mã PIN).

2.3. Vấn đề xác thực với OAuth2

OAuth 2.0 là một khung ủy quyền được phát triển cho web và do đó nó phải đối mặt với các mối đe dọa tương tự như tất cả các ứng dụng web. Để hiểu rõ hơn về những mối đe dọa này, hầu hết điểm yếu bảo mật ứng dụng web quan trọng được trình bày trong danh sách 10 OWASP hàng đầu, sau này được gọi đơn giản là OWASP T10. Thông qua việc xử lý các điểm yếu bảo mật OAuth 2.0 là có sẵn trong mô hình mối đe dọa OAuth 2.0

Điểm yếu đầu tiên trên OWASP T10, A1, là "Injection". "Các lỗi tiêm, chẳng hạn như SQL, OS, và việc đưa vào LDAP xảy ra khi dữ liệu không đáng tin cậy được gửi đến trình thông dịch như một phần của kết hợp ủy nhiệm hoặc truy vấn. Dữ liệu thù địch của kẻ tấn công có thể đánh lừa trình thông dịch thực thi ngoài ý muốn lệnh hoặc truy cập dữ liệu mà không có sự cho phép thích hợp." OAuth 2.0 yêu cầu dữ liệu đầu vào cả từ máy khách bên thứ ba cũng như từ người dùng. Một vài cái dữ liệu có thể được bao gồm trong các truy vấn cơ sở

dữ liệu, điều này có thể dẫn đến việc đưa dữ liệu vào trừ khi dữ liệu đó được vệ sinh đúng cách và thoát ra ngoài. Ngoài ra, tùy thuộc vào việc triển khai, các mã thông báo có thể cần mã hóa và giải mã. Tương tự như phân tích cú pháp, nó dễ bị tấn công tiêm.

OAuth 2.0 cung cấp một phương thức xác thực. Ngoài ra, máy chủ ủy quyền OAuth 2.0 có khả năng để sử dụng một số loại quản lý phiên khi xác thực chủ sở hữu tài nguyên. Một thích hợp cần triển khai để giảm thiểu rủi ro A2, "Xác thực bị hỏng và sự quản lý ". Các chức năng ứng dụng liên quan đến xác thực và quản lý phiên thường không được triển khai chính xác, cho phép kẻ tấn công xâm phạm mật khẩu, khóa hoặc phiên mã thông báo hoặc khai thác các lỗi triển khai khác để giả định danh tính của người dùng khác." Nếu quá trình triển khai OAuth 2.0 có sai sót, một số hoặc tất cả các mối đe dọa này có thể đã nhận ra.

Rủi ro A3, "Cross-Site Scripting (XSS)", có khả năng xảy ra khi chủ sở hữu tài nguyên ủy quyền màn hình, trừ khi dữ liệu được hiển thị cho người dùng, chẳng hạn như phạm vi được yêu cầu, được xác thực đúng cách. "Lỗi XSS xảy ra bất cứ khi nào một ứng dụng lấy dữ liệu không đáng tin cậy và gửi đến trình duyệt web mà không có xác nhận thích hợp hoặc thoát. XSS cho phép kẻ tấn công thực thi các tập lệnh trong nạn nhân trình duyệt có thể chiếm quyền điều khiển phiên của người dùng, phá hoại trang web hoặc chuyển hướng người dùng đến phần mềm độc hại các trang web. "

OAuth 2.0 thường được sử dụng để bảo vệ các tài nguyên web có một mã định danh tài nguyên đồng nhất (URI) có sẵn công khai hoặc tương đối dễ đoán. Do đó nó đặc biệt quan trọng để giảm thiểu điểm yếu A4, "Tham chiếu đối tượng trực tiếp không an toàn". "Đối tượng trực tiếp tham chiếu xảy ra khi một nhà phát triển để lộ tham chiếu đến một đối tượng triển khai nội bộ, chẳng hạn như tệp, thư mục hoặc khóa cơ sở dữ liệu. Nếu không có kiểm tra kiểm soát truy cập hoặc biện pháp bảo vệ khác, kẻ tấn công có thể thao túng các tham chiếu này để truy cập dữ liệu trái phép." Khi các tham chiếu đối tượng trực tiếp có sẵn và không thể sử dụng cho mỗi người dùng hoặc phiên trong- tham chiếu đối tượng trực tiếp, máy chủ tài nguyên sử dụng OAuth 2.0 phải kiểm tra quyền truy cập thích hợp mỗi khi chúng phục vụ các tài nguyên được bảo vệ.

Điểm yếu bảo mật A5, "Cấu hình sai bảo mật" là trọng tâm của nghiên cứu này. Bảo mật tốt yêu cầu có cấu hình an toàn được xác định và triển khai cho ứng dụng, khuôn khổ, máy chủ ứng dụng, máy chủ web, máy chủ cơ sở dữ

liệu và nền tảng. Cài đặt bảo mật nên được xác định, triển khai và duy trì, vì các giá trị mặc định thường không an toàn. Ngoài ra, phần mềm phải được cập nhật. Mô hình của chúng tôi nhằm mục đích xác định một mô hình triển khai bao gồm cấu hình an toàn cho OAuth 2.0 để cho phép nhà phát triển để tránh cấu hình sai bảo mật.

OAuth 2.0 giới thiệu thông tin xác thực, mã ủy quyền và mã thông báo bảo mật mới cho ứng dụng - cation. Những thông tin xác thực này cần được bảo vệ khỏi rủi ro A6, "Phơi nhiễm dữ liệu nhạy cảm".

"Nhiều ứng dụng web không bảo vệ đúng cách dữ liệu nhạy cảm, chẳng hạn như thẻ tín dụng, ID thuế, và thông tin xác thực. Những kẻ tấn công có thể đánh cắp hoặc sửa đổi dữ liệu được bảo vệ yếu kém như vậy.

Dữ liệu nhạy cảm cũng cần được bảo vệ thêm, chẳng hạn như mã hóa ở trạng thái nghỉ hoặc khi đang chuyển tiếp như các biện pháp phòng ngừa đặc biệt khi trao đổi với trình duyệt."

Mặc dù triển khai OAuth 2.0 không có giao diện người dùng (UI) lớn, nhưng điều quan trọng là để đảm bảo rằng một số yêu cầu được tạo từ giao diện người dùng được kiểm tra chính xác để giảm thiểu cổng A7, "Thiếu điều khiển truy cập mức chức năng". "Hầu hết các ứng dụng web chức năng xác minh cấp quyền truy cập trước khi hiển thị chức năng đó trong giao diện người dùng. Tuy nhiên, các ứng dụng cần thực hiện các kiểm tra kiểm soát truy cập giống nhau trên máy chủ khi mỗi chức năng là đã ngừng. Nếu các yêu cầu không được xác minh, những kẻ tấn công sẽ có thể giả mạo các yêu cầu để truy cập chức năng mà không có ủy quyền thích hợp. " Trong một số luồng ủy quyền OAuth 2.0, ủy quyền xảy ra trong tác nhân người dùng.

Điều quan trọng là các luồng ủy quyền này sử dụng mã thông báo bảo mật trong màn hình ủy quyền, để giảm thiểu mối đe dọa A8. "Truy vấn yêu cầu trên nhiều trang web (CSRF)". "Lực lượng tấn công CSRF trình duyệt của nạn nhân đã đăng nhập để gửi một yêu cầu HTTP giả mạo, bao gồm cả phiên của nạn nhân cookie và bất kỳ thông tin xác thực nào khác được tự động đưa vào một trang web để bị tấn công ứng dụng. Điều này cho phép kẻ tấn công buộc trình duyệt của nạn nhân tạo ra các yêu cầu ứng dụng để bị tấn công cho rằng đó là những yêu cầu hợp pháp từ nạn nhân. ".

Điểm yếu A9, "Sử dụng các thành phần có lỗi hỏng đã biết", không liên quan trực tiếp đến triển khai OAuth 2.0, nhưng thay vì xem xét bảo mật chung hơn. "Các thành phần, chẳng hạn như thư viện, khuôn khổ và các mô-đun phần

mềm khác, hầu như luôn chạy với đầy đủ chân. Nếu một thành phần dễ tổn thương bị khai thác, một cuộc tấn công như vậy có thể tạo điều kiện cho việc mất dữ liệu nghiêm trọng hoặc tiếp quản máy chủ. Các ứng dụng sử dụng các thành phần có lỗ hổng đã biết có thể làm suy yếu ứng dụng phòng thủ và kích hoạt một loạt các cuộc tấn công và tác động có thể xảy ra."

Một số loại cấp OAuth 2.0 sử dụng chuyển hướng như một phương tiện để đưa người dùng quay lại bên thứ ba khách hàng sau khi ủy quyền. Điều này làm cho OAuth 2.0 dễ bị ảnh hưởng bởi mối đe dọa A10 "

Các ứng dụng web thường chuyển hướng và chuyển tiếp người dùng sang các trang web người dùng khác, đồng thời sử dụng dữ liệu không đáng tin cậy để xác định các trang đích. Không có xác thực thích hợp, những kẻ tấn công có thể chuyển hướng nạn nhân đến các trang web lừa đảo hoặc phần mềm độc hại hoặc sử dụng chuyển tiếp để truy cập các trang trái phép.

Nhìn chung, 9/10 điểm yếu bảo mật ứng dụng web quan trọng nhất đã được trình bày trong OWASP T10 liên quan trực tiếp đến bất kỳ triển khai OAuth 2.0 nào. Đây chỉ là một phần các điểm yếu và mối đe dọa mà một nhà phát triển web triển khai OAuth 2.0 sẽ có thể xem xét và giảm thiểu để triển khai OAuth 2.0 một cách an toàn.

2.4. Tổng kết chương 2

Trong chương này tập trung tìm hiểu định nghĩa của OAuth2 và SSO, đồng thời nêu ra được một số tính năng bảo mật của OAuth2 và cơ chế hoạt động của SSO. Tuy nhiên vấn đề xác thực với OAuth2 là điều không tránh khỏi, chính vì thế trong chương này có liệt kê ra được một số điểm yếu quan trọng. Trong chương 2 chúng ta chỉ đưa ra tổng quan về giải pháp xác thực với OAuth2. Vì vậy để hiểu rõ về mô hình cũng như quy trình triển khai mô hình xác thực, chúng ta sẽ đi tìm hiểu ở chương 3.

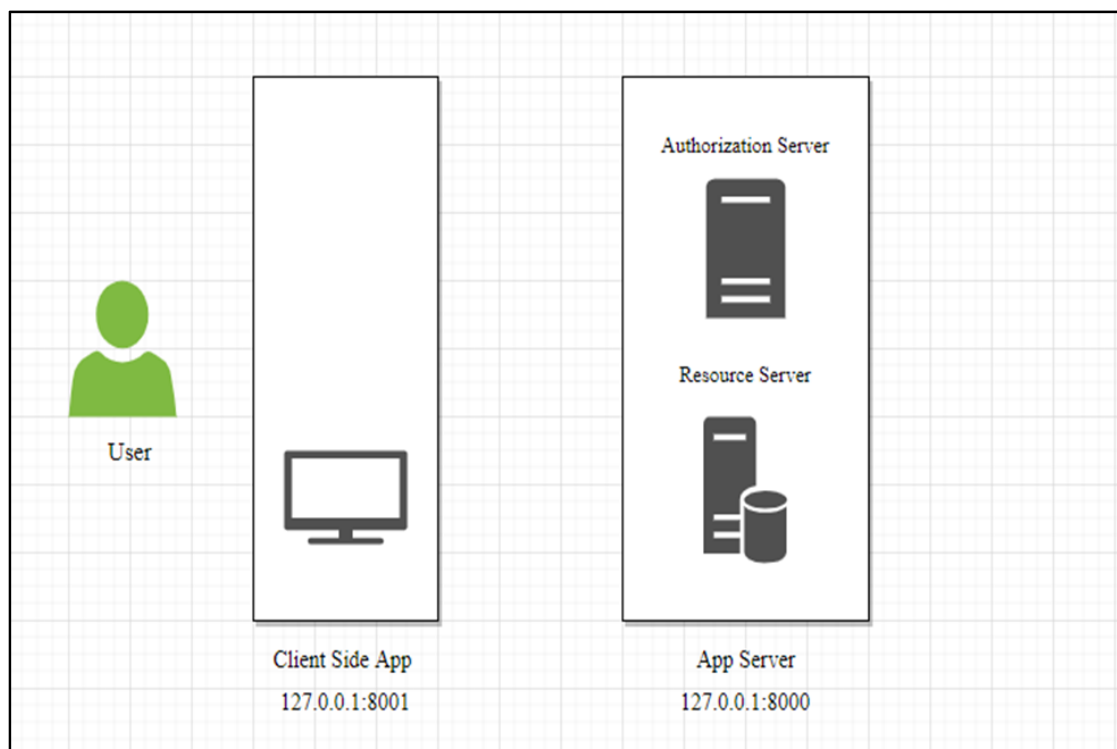
CHƯƠNG 3. ỨNG DỤNG TRIỂN KHAI MÔ HÌNH XÁC THỰC TẬP TRUNG

3.1. Mô hình triển khai

Mục đích của việc triển khai mô hình xác thực tập trung là cho phép người dùng có thể truy cập vào nhiều trang web, ứng dụng mà chỉ cần đăng nhập một lần duy nhất. Bài toán này cung cấp một giải pháp xác thực tập trung với OAuth2, giúp giảm nguy cơ lộ thông tin cá nhân, hỗ trợ trải nghiệm người dùng tốt hơn cho các ứng dụng gốc, mở rộng giao thức để cung cấp khả năng tương thích với các yêu cầu thiết bị trong tương lai, thuận tiện cho việc phát triển hệ thống sau này.

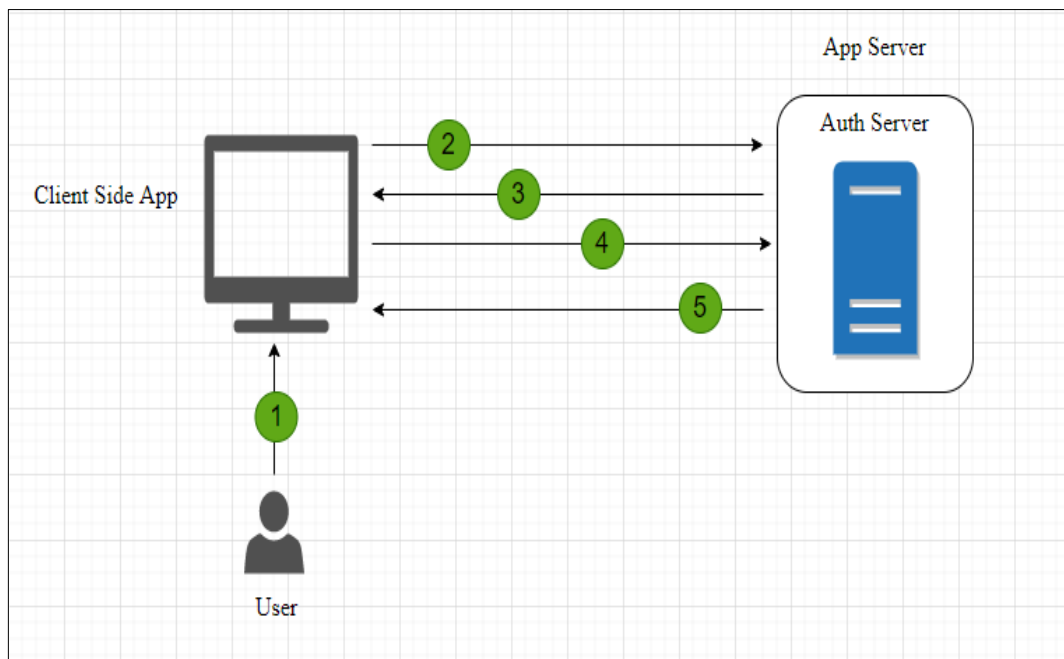
Mô hình triển khai sẽ như sau:

- App Server: Sử dụng để ủy quyền và cung cấp tài nguyên
- Client Side App: Ứng dụng khách, là ứng dụng cần được sự cho phép của người dùng để truy cập vào tài khoản.
- User: Làm chủ sở hữu tài nguyên, sử dụng để cấp quyền truy cập.



Hình 3.1. Mô hình triển khai xác thực

Mô hình luồng hoạt động Single Sign-On



Hình 3.2. Mô hình luồng Single Sign-On

- 1: User truy cập vào ứng dụng Client Side App.
- 2: Thực hiện đăng nhập hoặc gửi yêu cầu ủy quyền đến App Server.
- 3: Khi App Server nhận được yêu cầu ủy quyền, thông qua máy chủ ủy quyền sẽ kiểm tra và gửi về cho Client Side App một mã ủy quyền “Auth_code”.
- 4: Sau khi Client Side App nhận được mã ủy quyền sẽ tiến hành gửi lên App Server một thông tin gồm: Auth_code, client_id, client_secret
- 5: App Server nhận được sẽ kiểm tra thông tin client_id, client_secret kèm theo mã ủy quyền, nếu hợp lệ thì trả về access_token và refresh_token cho Client Side App. Khi đó Client Side App đã có được access_token và refresh_token do đó nó biết được người dùng đã xác thực thành công.

3.2. Quy trình triển khai

Như trong mô hình triển khai xác thực, chúng ta sẽ xây dựng 2 hệ thống là App Server chính là nơi cung cấp các dịch vụ API, bao gồm máy chủ ủy quyền và máy chủ cung cấp tài nguyên. Client Side App chính là ứng dụng khách. Chúng ta sẽ tạo ra hai project laravel cho 2 hệ thống lần lượt là “student_managment” và “website_client”. Dưới đây là quá trình xây dựng hệ thống.

3.2.1. Xây dựng App Server

- ❖ Khởi tạo project có tên là “student_managerment” sử dụng laravel

```
composer create-project --prefer-dist laravel/laravel:^7.*  
student_managerment
```

- ❖ Cài đặt gói laravel passport

```
composer require laravel/passport "~9.0"
```

- ❖ Đăng ký provider trong config/app.php

```
Laravel\Passport\PassportServiceProvider::class
```

- ❖ Kết nối database trong tệp .env. Tìm dòng “DB_DATABASE” và chỉnh sửa database thành “student_management”.

```
DB_CONNECTION=mysql  
DB_HOST=127.0.0.1  
DB_PORT=3306  
DB_DATABASE=student_management  
DB_USERNAME=root  
DB_PASSWORD=
```

- ❖ Chạy lệnh migrate để tạo các bảng trong database

```
php artisan migrate
```

- ❖ Chạy lệnh tạo client sẽ tạo ra client id và client secret được dùng để tạo các mã truy cập.

```
php artisan passport:install
```


- ❖ Thêm dòng “Laravel\Passport\HasApiTokens” vào trong app\User model và tìm đến class User thêm vào dòng “use HasApiTokens”.

```
<?php
// app/User.php
namespace App;
use Laravel\Passport\HasApiTokens;
use Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable
{
    use HasApiTokens, Notifiable;
```

- ❖ Đăng ký route mặc định cho Laravel Passport trong app/Providers/AuthServiceProvider.php. Tìm đến hàm “boot” thêm vào dòng “Passport::routes()”.

```
<?php
namespace App\Providers;
use Illuminate\Foundation\Support\Providers\AuthServiceProvider as
ServiceProvider;
use Illuminate\Support\Facades\Gate;
use Laravel\Passport\Passport;
class AuthServiceProvider extends ServiceProvider
{
    protected $policies = [
        // 'App\Model' => 'App\Policies\ModelPolicy',
    ];
    public function boot()
    {
        $this->registerPolicies();
        Passport::routes();
    }
}
```

- ❖ Thiết lập driver cho api trong config/auth.php. Tìm dòng “driver”, thay “token” bằng “passport”.

```
'guards' => [  
    'web' => [  
        'driver' => 'session',  
        'provider' => 'users',  
    ],  
  
    'api' => [  
        'driver' => 'passport',  
        'provider' => 'users',  
        'hash' => false,  
    ],  
],
```

- ❖ Publish Vue component xây dựng sẵn bằng cách chạy lệnh bên dưới

```
php artisan vendor:publish --tag=passport-components
```

❖ Khai báo các component vào tệp resource/js/app.js

```
Vue.component(
  'passport-clients',
  require('./components/passport/Clients.vue').default
);

Vue.component(
  'passport-authorized-clients',
  require('./components/passport/AuthorizedClients.vue').default
);

Vue.component(
  'passport-personal-access-tokens',
  require('./components/passport/PersonalAccessTokens.vue').default
);
```

❖ Chạy lệnh bên dưới để biên dịch tài nguyên sử dụng Laravel Mix

```
npm install
npm run dev
```

❖ Cài đặt Laravel Authentication

```
composer require laravel/ui:^2.4
php artisan ui bootstrap --auth
php artisan ui vue --auth
npm install
npm run dev
```

3.2.2. Xây dựng Client Side App

- ❖ Tương tự như phần xây dựng App Server, chúng ta tạo ra project “website_client” sử dụng laravel

```
composer create-project --prefer-dist laravel/laravel:^7.*  
website_client
```

- ❖ Cài đặt gói Guzzle HTTP. Guzzle là một PHP HTTP client giúp việc gửi HTTP request trở lên đơn giản hơn.

```
composer require guzzlehttp/guzzle
```

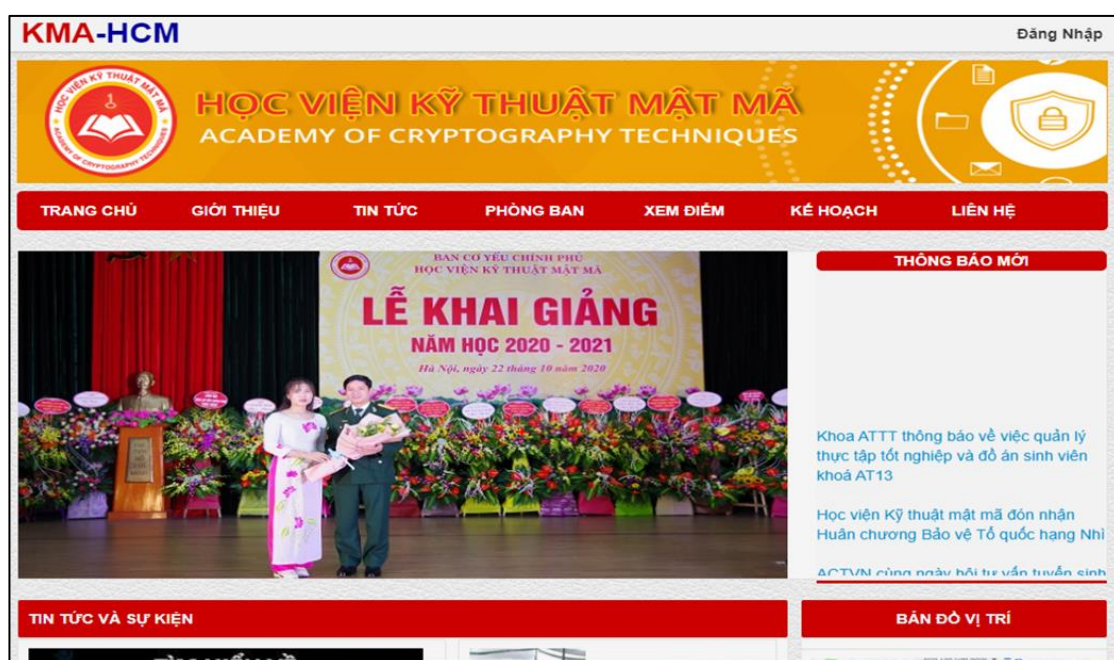
3.2.3. Thực hiện cài đặt xác thực

Chúng ta cùng thực hiện xác thực, cấp quyền bằng mã ủy quyền giống như khi tích hợp đăng nhập vào Facebook, hay Google, Twiter..., ở Client Side App (là project website_client) sẽ thực hiện tích hợp đăng nhập cấp bởi App Server (là project student_managment)

3.2.3.1. Thực hiện đăng ký ứng dụng ở App Server

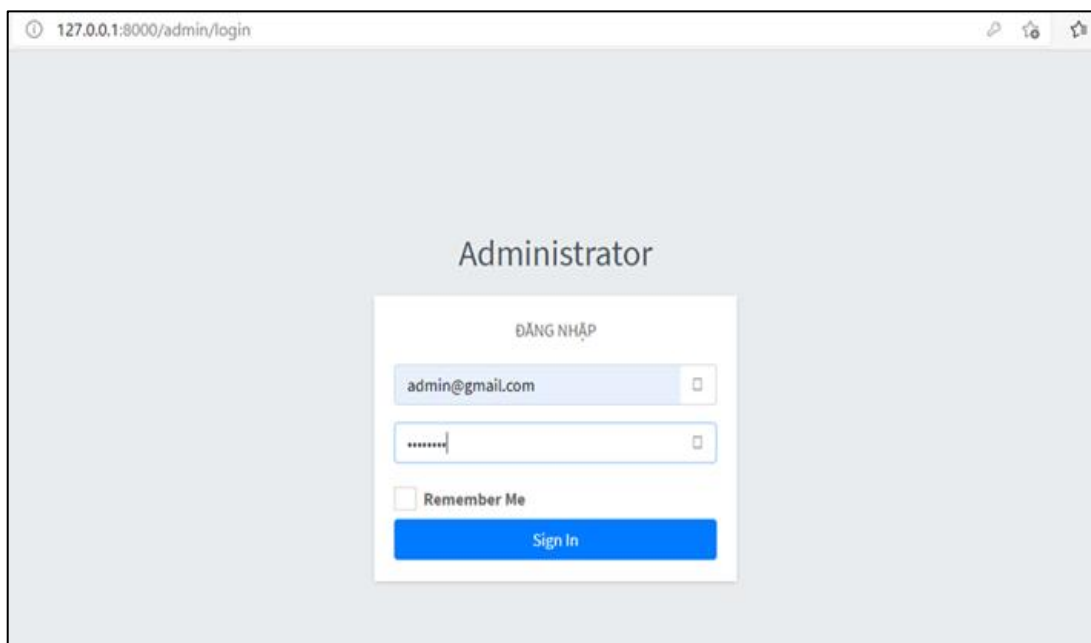
Trước khi sử dụng OAuth2 cho ứng dụng, bạn phải đăng ký ứng dụng với bên cung cấp dịch vụ. Việc đăng kí ứng dụng giống như vào các trang dành cho nhà phát triển.

❖ Bước 1: Truy cập vào giao diện của App Server



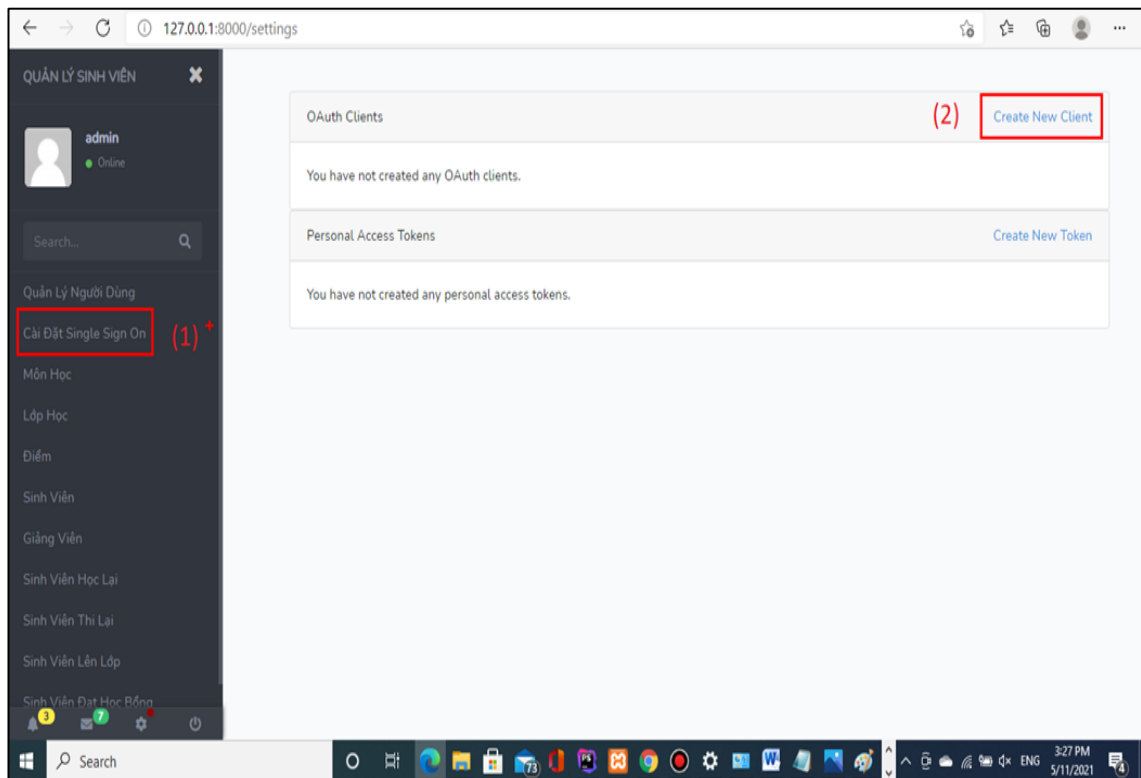
Hình 3.3. Giao diện của App Server

❖ Bước 2: Đăng nhập bằng tài khoản quản trị



Hình 3.4. Tài khoản quản trị

❖ Bước 3: Chuyển đến trang quản trị để thực hiện cấu hình OAuth2 – Passport Single Sign-On như sau:



Hình 3.5. Trang quản trị

- 1: Click vào để thực hiện cài đặt OAuth2
 - 2: Click vào để đăng ký ứng dụng khách, ứng dụng sẽ được xác thực bởi Server này.
- ❖ Bước 4: Nhập các thông tin đăng kí vào phần đăng ký
- 1: Nhập application name: là tên của ứng dụng
 - 2: Nhập redirect chính là địa chỉ sẽ quay về sau khi quá trình ủy quyền hoàn tất (cho phép hoặc từ chối từ phía User), chính vì thế mà địa chỉ quay về đó chính là nơi bạn sẽ phải thực hiện xử lý cho authorization codes hoặc access tokens.

- 3: Click vào create sẽ tạo ra ứng dụng khách

Create Client [X]

Name: (1)
Something your users will recognize and trust.

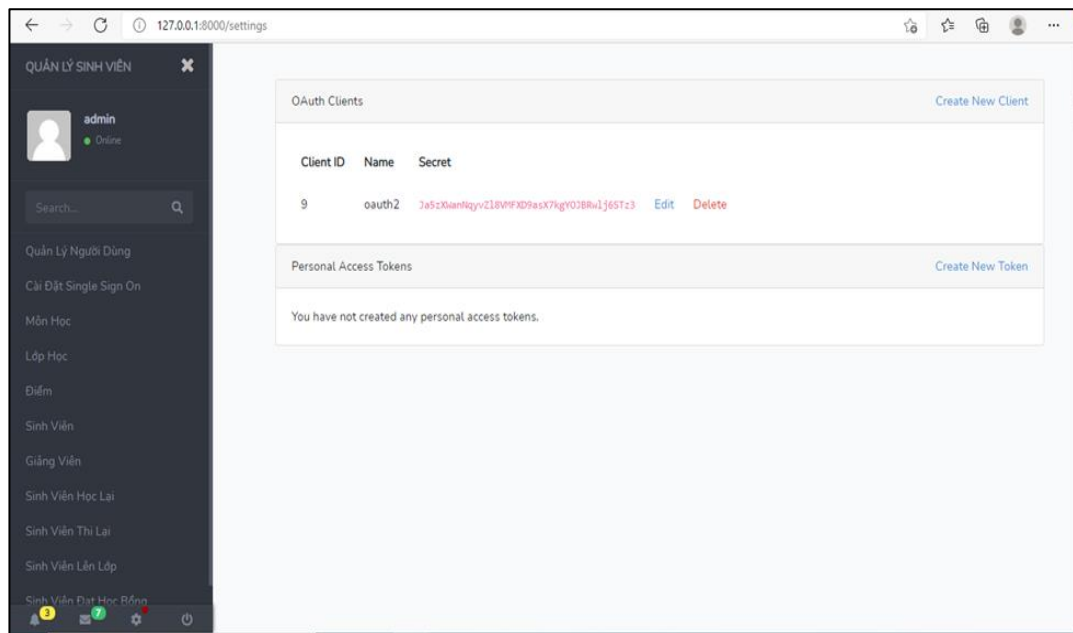
Redirect URL: (2)
Your application's authorization callback URL.

Confidential: ☒
Require the client to authenticate with a secret. Confidential clients can hold credentials in a secure way without exposing them to unauthorized parties. Public applications, such as native desktop or JavaScript SPA applications, are unable to hold secrets securely.

[Close] [Create] (3)

Hình 3.6. Nhập thông tin đăng ký

- ❖ Bước 5: Khi ứng dụng của bạn được đăng ký, bên dịch vụ sẽ phát hành "thông tin chứng thực client" (client credentials) cho bạn bao gồm thông tin:
 - Client Identifier: là một chuỗi ký tự được sử dụng bởi Service API để định danh ứng dụng, đồng thời cũng được dùng để xây dựng "authorization URL" hiển thị phía User.
 - Client Secret: là một chuỗi ký tự được sử dụng để xác thực định danh (ID) của ứng dụng khi ứng dụng yêu cầu truy cập thông tin tài khoản của User. Chuỗi này được giữ bí mật giữa Application và API.



Hình 3.7. Client Identifier và Client Secret

3.2.3.2. Thực hiện xây dựng callback ở Client Side App

- ❖ Bước 1: Xây dựng hàm connection: Mục đích của hàm này được dùng để lấy mã truy cập từ App Server, gồm các trường:

```
public function connection(Request $request) {
    $request->session()->put('state', $state = Str::random(40));
    $query = http_build_query([
        'client_id' => "9",
        'redirect_uri' => 'http://127.0.0.1:8001/callback',
        'response_type' => 'code',
        'scope' => "",
        'state' => $state
    ]);
    return redirect('http://127.0.0.1:8000/oauth/authorize?'.$query);
}
```

- 'client_id'='9': Giá trị 9 là Client ID của Client Side App sau khi đăng ký với App Server
- 'redirect_uri'=' http://127.0.0.1:8001/callback': là nơi thực hiện các xử lý cho authorization_code
- 'response_type'='code': tham số để hiểu được ứng dụng đang yêu cầu nhận authorization_code.

- 'Scope'='' trường này định nghĩa phạm vi cho phép Client Side App truy cập.
- 'state'=\$state, \$state là giá trị session
- “redirect('http://127.0.0.1:8000/oauth/authorize?'.\$query)” là nơi được chuyển hướng đến để lấy mã truy cập (App Server)

❖ Bước 2: Xây dựng hàm callback

```
public function callback(Request $request) {
    $http = new Client();
    $response = $http->post( 'http://127.0.0.1:8000/oauth/token', [
        'form_params' => [
            'grant_type' => 'authorization_code',
            'client_id' => '9',
            'client_secret' => 'Ja5zXWanNqyvZl8VMFXD9asX7kgYOJBRwlj6STz3',
            'redirect_uri' => 'http://127.0.0.1:8001/callback',
            'code' => $request->code,
        ]
    ]);
    $request->session()->put('token', json_decode((string)$response->getBody(), true));
    $body = json_decode((string) $response->getBody(), true);
    $response = $http->get('http://127.0.0.1:8000/api/user', [
        'headers' => [
            'Authorization' => 'Bearer ' . $body['access_token'],
        ],
    ]);
    $user = json_decode((string) $response->getBody(), true);

    return view('welcome')->with([
        'user' => $user
    ]);
}
```

Thực hiện việc ủy quyền, xác thực, chính là đoạn code này:

```
$response = $http->post( 'http://127.0.0.1:8000/oauth/token', [
    'form_params' => [
        'grant_type' => 'authorization_code',
        'client_id' => '9',
        'client_secret'=>
'Ja5zXWanNqyvZl8VMFXD9asX7kgYOJBRwlj6STz3',
        'redirect_uri' => 'http://127.0.0.1:8001/callback',
        'code' => $request->code,
    ]
]);
```

- ‘grant_type’ => ‘authorization_code’ là loại ủy quyền được sử dụng
- ‘client_id’=> ‘9’, tương tự ở hàm connection là Client ID của Client Side App sau khi đăng ký với App Server
- ‘client_secret’=>‘Ja5zXWanNqyvZl8VMFXD9asX7kgYOJBRwlj6STz3’, là chuỗi được sử dụng để xác thực định danh (ID) của ứng dụng khi ứng dụng yêu cầu truy cập, chuỗi này được tạo ra khi đăng kí ở App Server.
- ‘redirect_uri’ => 'http://127.0.0.1:8001/callback' là nơi thực hiện các xử lý cho access token
- ‘code’ => \$request->code, cơ chế mã hóa được sử dụng

Thực hiện lấy thông tin người dùng sau khi xác thực thành công. Chúng ta thực hiện gửi GET đến “http://127.0.0.1:8000/api/user” với header có dạng Authorization: Bearer AUTHORIZATION_CODE

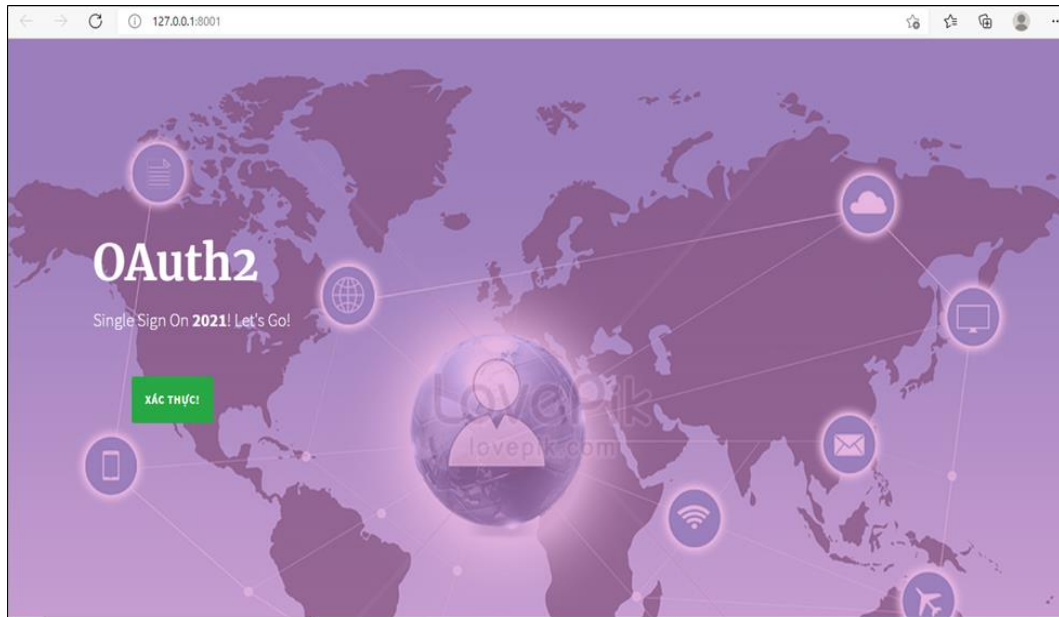
```
$request->session()->put('token', json_decode((string)$response->getBody(), true));  
$body = json_decode((string) $response->getBody(), true);  
$response = $http->get('http://127.0.0.1:8000/api/user', [  
    'headers' => [  
        'Authorization' => 'Bearer ' . $body['access_token'],  
    ],  
]);  
$user = json_decode((string) $response->getBody(), true);
```

- ❖ Bước 3: Đăng ký route vào routes/web.php: được sử dụng để xử lý các hành động khi gọi đến các hàm connection và callback mà ta đã xây dựng ở các bước trên

```
Route::get('/connection', 'Oauth2SettingController@connection')->name('connection');  
Route::get('/callback', 'Oauth2SettingController@callback')->name('callback');
```

3.3. Kết quả thực nghiệm

- ❖ Bước 1: Người dùng truy cập vào ứng dụng Client Side App



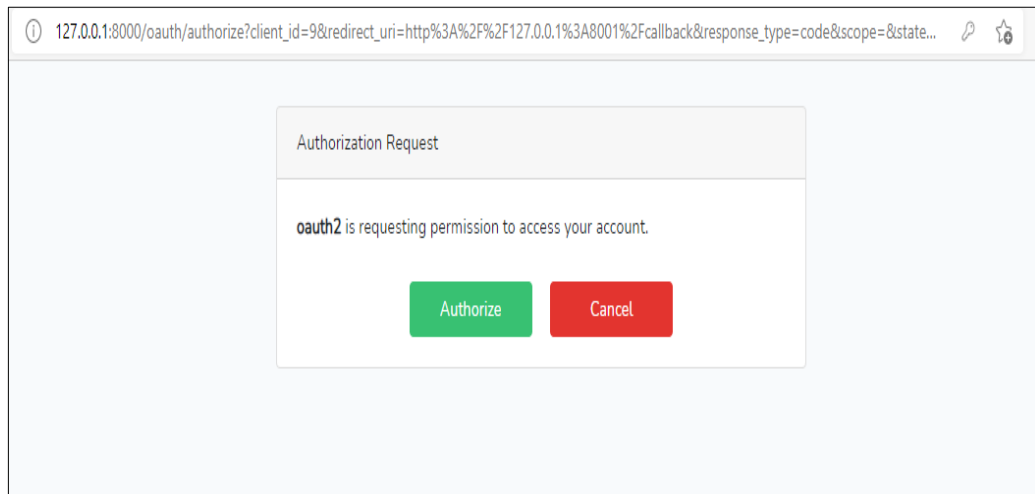
Hình 3.8. Giao diện Client Side App

- ❖ Bước 2: Khi người dùng click vào “XÁC THỰC” thì ứng dụng Client Side App sẽ chuyển hướng người dùng đến App Server để bắt đầu quá trình nhận authorization code. Người dùng được chuyển đến trang đăng nhập. Người dùng nhập thông tin đăng nhập để xác thực.

Hình 3.9. Đăng nhập xác thực

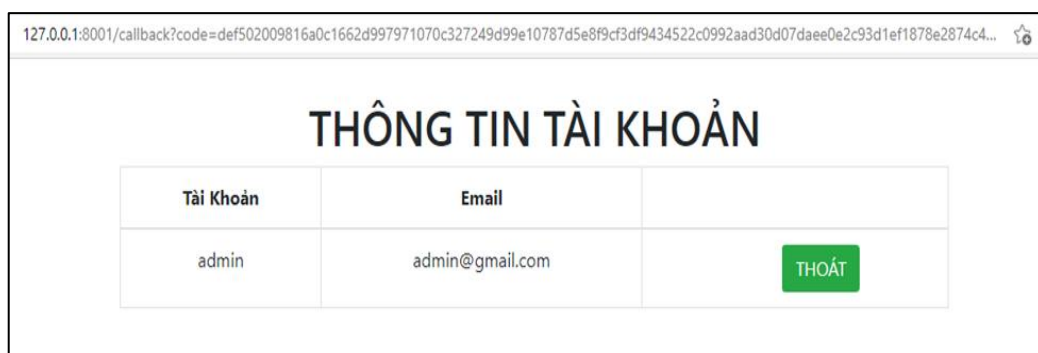
- ❖ Bước 3: Sau khi đăng nhập xong “App Server” sẽ xác thực thông tin đăng nhập và chuyển hướng người dùng đến "redirect uri" của ứng dụng (nơi ứng dụng bắt thông tin trả về từ App Server) kèm theo một đoạn

"authorization code" và hiện ra thông báo người dùng cần ủy quyền cho ứng dụng Client Side App (Giống như khi thực hiện với Facebook, Google...).



Hình 3.10. Thực hiện ủy quyền

- ❖ Bước 4: Người dùng bấm vào Authorize để ủy quyền cho ứng dụng. App Server sẽ kiểm tra thông tin Client Id, Client Secret kèm theo mã ủy quyền, nếu xác thực thành công thì trả về access token và refresh token cho Client Side App. Khi đó Client Side App đã có được access token và refresh token. Do đó nó biết được người dùng đã xác thực thành công. Khi đó thì có thể lấy được thông tin người dùng.



Hình 3.11. Xác thực thành công

3.4. Kết luận chương 3

Trong chương 3, đồ án trình bày chi tiết công việc xây dựng mô hình, triển khai xác thực tập trung với OAuth2 để xác thực một trang web. OAuth2 làm cho việc tích hợp Single Sign-On vào các ứng dụng mới hoặc đã có trở nên dễ dàng hơn nhiều và được xem như một giải pháp tiêu chuẩn để xác thực, phân quyền giữa các ứng dụng khác nhau. OAuth2 hỗ trợ việc nhiều loại xác thực khác nhau, tách biệt vai trò của việc nhận ủy quyền người dùng và xử lý các cuộc gọi API. Đối với các nhà cung cấp lớn hơn cần khả năng mở rộng này và các nhà cung cấp nhỏ hơn có thể sử dụng cùng một máy chủ cho cả hai vai trò.

KẾT LUẬN

Các hệ thống phân tán ngày càng trở lên phổ biến và xác thực là một khía cạnh quan trọng của tất cả chúng. Single Sign-On giải quyết một vấn đề lớn: làm thế nào để quản lý được số lượng người dùng đang tăng lên trên toàn bộ hệ thống gồm nhiều ứng dụng và dịch vụ. Giải pháp xác thực tập trung giúp:

- Giảm số lần đăng nhập vào hệ thống, rất tiện lợi cho người dùng chỉ cần đăng nhập một lần duy nhất và sử dụng hệ thống ứng dụng.
- Không cần ghi nhớ nhiều tài khoản và mật khẩu.
- Hạn chế nguy cơ lộ thông tin cá nhân.
- Giúp việc phát triển hệ thống ứng dụng được đơn giản hóa.

Vấn đề còn tồn tại

- Bên cạnh các kết quả đã đạt được vẫn còn một số hạn chế cần khắc phục trong triển khai giải pháp như:
 - Chỉ tập trung vào triển khai giải pháp chưa đi sâu vào các tiêu chuẩn bảo mật, lỗ hổng và biện pháp phòng tránh trong giao thức OAuth2.
 - Xác thực và lấy thông tin người dùng chỉ dừng lại ở mức độ lấy ra thông tin tài khoản người dùng.

Hướng phát triển kế tiếp

- Với những hạn chế đã phân tích ở trên, chúng tôi sẽ tiếp tục thực hiện nghiên cứu và hoàn thiện mô hình giải pháp
 - Vấn đề bảo mật: Tìm hiểu các lỗ hổng cấp mã xác thực, vấn đề xác thực chuyển hướng URL, rò rỉ các thông tin xác thực qua header hoặc lịch sử trình duyệt,...
 - Thực hiện xác thực và lấy ra nhiều thông tin hơn.

TÀI LIỆU THAM KHẢO

- [1]. Tìm hiểu về OAuth2 của Viblo: <https://viblo.asia/p/understanding-oauth2-aWj53L31K6m>
- [2]. Cài đặt project laravel: <https://laravel.com/docs/7.x/installation>
- [3]. Cài đặt và sử dụng Passport: <https://laravel.com/docs/7.x/passport#introduction>
- [4]. Tìm hiểu Guzzle: <https://viblo.asia/p/tao-request-httpclient-toi-server-that-don-gian-voi-guzzle-bJzKmxQO59N>
- [5]. Bài viết về các loại ủy quyền trong OAuth2: <https://developer.okta.com/blog/2018/04/10/oauth-authorization-code-grant-type>
- [6]. Bài viết hoạt động Single Sign-On: <https://auth0.com/blog/what-is-and-how-does-single-sign-on-work/>
- [7]. Các vấn đề bảo mật OAuth2: <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-security-topics>
- [8]. Cài đặt máy chủ OAuth2 sử dụng Laravel Passport: <https://blog.shameerc.com/2016/08/set-up-oauth2-server-using-laravel-passport>
- [9]. Tìm hiểu về SSO của Nick Heijmink: https://www.ru.nl/publish/pages/769526/z_researchpaper_sso_final_nick_heijmink_s4250559.pdf
- [10]. Tìm hiểu về Oauth2 của Ari-Pekka Koponen: <https://jyx.jyu.fi/bitstream/handle/123456789/51065/1/URN%3ANBN%3Afi%3Aju-201608253883.pdf>