ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

NGUYỄN HỒNG SƠN
NGUYỄN HẢI LONG

KHÓA LUẬN TỐT NGHIỆP

MÃ HÓA VÀ KIỂM SOÁT QUYỀN TRUY CẬP TRÊN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU SQL

ENCRYPT AND ACCESS CONTROL ON SQL DATABASE MANAGEMENT SYSTEM

KỸ SƯ NGÀNH AN TOÀN THÔNG TIN

TP. HỒ CHÍ MINH, NĂM 2021

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

NGUYỄN HỒNG SƠN – 17520988
NGUYỄN HẢI LONG – 17520712

KHÓA LUẬN TỐT NGHIỆP

MÃ HÓA VÀ KIỂM SOÁT QUYỀN TRUY CẬP TRÊN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU SQL

ENCRYPT AND ACCESS CONTROL ON SQL DATABASE MANAGEMENT SYSTEM

KỸ SƯ NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN
TS. NGUYỄN NGỌC TỰ

TP. HỒ CHÍ MINH, 2021

THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số 463/QĐ-ĐHCNTT ngày 23 tháng 7 năm 2021 của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1.	TS. Nguyễn Tuấn Nam	Chủ tịch
2.	ThS. Nguyễn Duy	Ủy viên
3.	ThS. Trần Hồng Nghi	Thư ký

LỜI CẢM ƠN

Không ai đạt được điều gì đó to lớn mà không nhờ sự giúp đỡ của những người xung quanh, cho dù là trực tiếp hay gián tiếp đi nữa. Để hoàn thành được khóa luận này, nhóm tác giả may mắn nhận được nhiều sự giúp đỡ và hỗ trợ từ quý thầy, cô, anh chị, bạn bè và người thân. Nhóm tác giả xin dành những trang đầu tiên này để bày tỏ lòng tri ân của mình tới tất cả mọi người, những người đã đồng hành cùng nhóm trong khoảng thời gian vừa qua.

Đầu tiên, nhóm tác giả xin gửi lời cảm ơn sâu sắc đến toàn thể các thầy cô của Trường Đại học Công nghệ Thông tin nói chung và các thầy cô khoa Mạng máy tính và Truyền thông nói riêng. Nhờ những kiến thức quý giá mà thầy cô đã truyền đạt, cũng như việc hỗ trợ tận tình trong suốt khoảng thời gian thực hiện, nhóm đã hoàn thành khóa luận và đạt được các kết quả đáng ghi nhận.

Nhóm tác giả xin đặc biệt cảm ơn TS. Nguyễn Ngọc Tự là người đã truyền cảm hứng, tận tình hướng dẫn và hỗ trợ tận tình về kiến thức, tạo môi trường thuận lợi để nhóm có thể học hỏi, trao đổi với các bạn, các em trong nhóm nghiên cứu. Đây là những kiến thức, kinh nghiệm quý giá, không chỉ có tác dụng trong khóa luận tốt nghiệp này mà còn trong khoảng thời gian làm việc trong chặng đường tiếp theo.

Trong giai đoạn dịch bệnh khó khăn, tình hình ngày càng phức tạp, dù có khó khăn trong nhiều công việc, nhóm nhận được nhiều sự giúp đỡ và động viên từ thầy cô và bạn bè. Đây là động lực to lớn thúc đẩy nhóm làm việc trong suốt quá trình tìm hiểu và hoàn thành khóa luận này.

Cuối cùng, nhóm tác giả không quên bày tỏ lòng tri ân đến gia đình và người thân, những người đã luôn là những hậu phương vững chắc và luôn ủng hộ từng quyết định mà nhóm đưa ra.

Mặc dù đã nỗ lực rất nhiều để luận văn được hoàn thiện nhất, song khó có thể tránh khỏi thiếu sót và hạn chế. Kính mong nhận được sự thông cảm và ý kiến đóng góp từ quý thầy cô và các bạn.

TP. Hồ Chí Minh, ngày 12 tháng 7 năm 2021

Nhóm tác giả

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

DANH MỤC TỪ VIẾT TẮT

ABAC	Attribute-based access control
BLP	Bell – La Padula
CSP	Cloud Service Provider
DAC	Discretionary access control
FsP	Forward search privacy
MAC	Mandatory access control
ORAM	oblivious RAM
PIR	private information retrieval
RBAC	Role-Based access control
SE	Searchable Encryption
SSE	Searchable Symmetric Encryption

TÓM TẮT KHÓA LUẬN

Vào năm 2008, Lizhe Wang, một trong những ngọn cờ tiên phong đã đưa ra khái niệm cơ bản về "Cloud Computing". Theo đó, điện toán đám mây là một tập hợp các dịch vụ hỗ trợ mạng, cung cấp khả năng mở rộng, chất lượng dịch vụ được đảm bảo, thường được cá nhân hóa, chi phí thấp theo yêu cầu và có thể truy cập một cách đơn giản và phổ biến [2].

Từ thời điểm đó, điện toán đám mây đã dần trở thành một trong những từ khóa tiếp theo của ngành công nghệ thông tin. Vào năm 2020, tổng giá trị của thị trường là 371,4 tỷ USD, tốc độ tăng trưởng kép hàng năm (CAGR) là 17,5%. Theo dự kiến, tới năm 2025, thị trường điện toán đám mây sẽ có giá trị tới 832,1 tỷ USD, sẽ có hơn 100 zettabytes dữ liệu được lưu trữ trên đám mây. Trong cùng khoảng thời gian đó, tổng dung lượng dữ liệu được lưu trữ sẽ vượt quá 200 zettabytes, nghĩa là có khoảng 50% dữ liệu được lưu trên đám mây, con số này là 25% vào năm 2015 [3].

Ở Việt Nam hiện nay, đón đầu xu thế trong cuộc cách mạng công nghệ 4.0, Chính phủ thúc đẩy mạnh mẽ và tạo điều kiện trên mọi phương diện nhằm thúc đẩy khởi nghiệp, nhóm ngành công nghệ thông tin là một trong những lĩnh vực được ưu tiên thúc đẩy phát triển. Nhiều công ty công nghệ thông tin được thành lập và có tiềm năng phát triển cao. Đi đôi với việc thành lập doanh nghiệp, các công ty xây dựng cơ sở dữ liệu của mình nhằm phục vụ các hoạt động nội bộ của công ty cũng như kinh doanh với đối tác, khách hàng. Xu hướng sử dụng cloud service để lưu và quản trị cơ sở dữ liệu được nhiều đơn vị lựa chọn.

Tuy có những ưu điểm vượt trội như trên, hệ thống này cũng đặt ra nhiều vấn đề về an toàn thông tin cho dữ liệu được lưu trữ trên đám mây. Môi trường đám mây được xem là không tin cậy cho những dữ liệu riêng tư có tính nhạy cảm: dữ liệu có thể bị truy cập trái phép từ nhà quản cung cấp dịch vụ, bị rò rỉ sang bên thứ ba không liên quan hoặc bị đánh cắp bởi quản trị viên, hacker. Các công ty cho thuê dịch vụ đám mây phục vụ nhiều khách hàng có thể dẫn tới việc không duy trì được sự tách biệt giữa những người

cùng thuê dịch vụ. Kẻ tấn công có thể sử dụng lỗi này để dành quyền truy cập từ tài nguyên của tổ chức này vào tài nguyên của tổ chức khác. Dữ liệu của người dùng sau khi hết sử dụng có thể không được xóa an toàn do bị giảm khả năng hiển thị vào nơi dữ liệu được lưu trữ vật lý trên đám mây hoặc bị mất mát do nhà cung cấp dịch vụ vô tình xóa hoặc thảm họa. Người dùng nội bộ của công ty cho thuê có thể lạm quyền và truy cập nhằm đánh cắp thông tin [4]. Do vậy, ngoài những cơ chế bảo mật sẵn có từ nhà cung cấp dịch vụ, chủ sở hữu dữ liệu cần có những cơ chế bảo mật riêng để đảm bảo thông tin lưu trữ, tương tác và trao đổi được an toàn. Một trong những giải pháp được sử dụng là mã hóa dữ liệu trước khi lưu trữ tại đám mây và thay đổi phương pháp tương tác truyền thống trên dữ liệu rõ bằng các phương pháp tương tác trên dữ liệu mã hóa.

Để thử nghiệm và tìm ra phương pháp mã hóa, kiểm soát truy cập với cơ sở dữ liệu được lưu trữ trên đám mây, nhóm tác giả đã tiến hành nghiên cứu và xây dựng mô hình nhằm mục đích đảm bảo sự an toàn của cơ sở dữ liệu trước những rủi ro của hệ thống điện toán đám mây. Từ đó, rút ra nhận xét và hướng phát triển trong tương lai.

CHƯƠNG 1. **GIỚI THIỆU**

1.1. Khái quát chung

Trong kỷ nguyên mà sự phát triển của các hệ thống điện toán đám mây càng phát triển với độ nở lớn như hiện nay, lượng dữ liệu đổ lên các dịch vụ đám mây đang tăng theo cấp số nhân. Đại dịch Covid-19 vẫn đang lan rộng, thành tựu về vắc – xin tuy đã kiểm soát khá tốt tốc độ lây lan của dịch bệnh, nhưng vẫn chưa thể xóa sổ hoàn toàn nó ra khỏi đời sống kinh tế, xã hội. Dịch bệnh thúc đẩy phát triển những cách làm việc phi truyền thống, chuyển dần sang nền tảng trực tuyến, nơi mà dòng chảy dữ liệu vốn đã chật chội nay càng thêm khó kiểm soát. Vào năm 2020, tổng chi tiêu của người dùng cuối cho các dịch vụ đám mây đạt tổng cộng 270 tỷ USD, con số này dự kiến sẽ tăng với mức đáng kinh ngạc là 23,1% vào năm 2021, lên 332,3 tỷ USD. Trong khi đó, 48% doanh nghiệp chọn lưu trữ dữ liệu quan trọng của họ, bao gồm dữ liệu mã hóa và dữ liệu thông thường trên đám mây. Vì vậy, không có gì ngạc nhiên khi 75% doanh nghiệp coi các vấn đề bảo mật đám mây là mối quan tâm hàng đầu. Trong số đó, 33% người được hỏi cực kỳ quan tâm, 42% cực kỳ lo ngại và 25% không quan tâm hoặc quan tâm vừa phải [3].

Vì nhiều công nghệ điện toán đám mây đang ngày càng được sử dụng nhiều hơn, và do đó, giống như hầu hết các công nghệ mới, vấn đề bảo mật cho nó đã, đang và tiếp tục được đặt ra và ngày càng trở nên quan trọng. Dữ liệu là nguồn sống của doanh nghiệp, và khi nó càng lớn thì động lực bảo vệ tài sản quý giá này thúc đẩy họ tìm kiếm các giải pháp bảo mật an toàn hơn. Vấn đề chính đặt ra là tìm một nhà cung cấp dịch vụ đám mây (CSP) có uy tín và ổn định để các doanh nghiệp có thể ít bị tấn công hơn hay bản thân các công ty phải chủ động phát triển một công cụ như là chìa khóa két sắt để bảo vệ tài sản của mình khi giao cho người khác nắm giữ.

Cả hai hướng tiếp cận này đều có những ưu và nhược điểm. Sử dụng dịch vụ của một đối tác CSP uy tín giúp chủ sở hữu dữ liệu tiết kiệm được nhiều kinh phí, mức độ tin cậy cũng được đảm bảo ở mức tương đối. Tuy nhiên, hướng tiếp cận này chỉ giải

quyết được một số vấn đề xảy ra từ những đối tượng xấu có ý đồ tấn công mà không giải quyết được các vấn đề có thể xảy ra từ bên cung cấp dịch vụ. Ngược lại, để phát triển một giải pháp bảo mật hiệu quả, nguồn lực kinh phí và con người phải bỏ ra là rất lớn. Không phải công ty, doanh nghiệp nào cũng có thể đầu tư nguồn lực lớn vào vấn đề này.

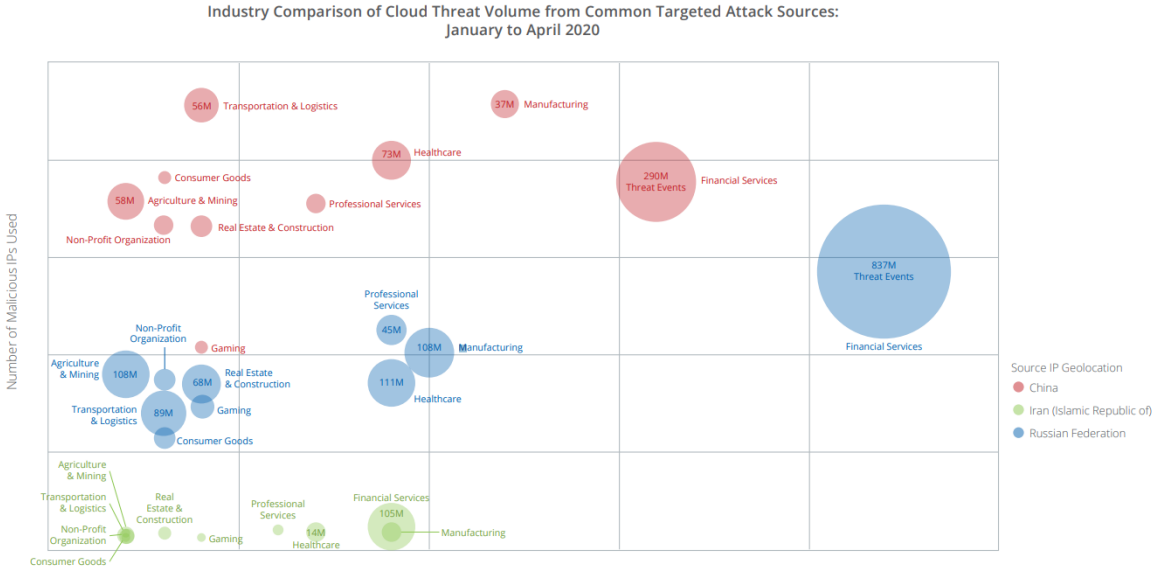
Nhằm khắc phục điểm yếu và phát huy điểm mạnh của hai hướng trên, việc một bên thứ ba có đầy đủ năng lực chuyên môn và nguồn lực tài chính đứng ra phát triển dịch vụ bảo mật trên nền tảng đám mây giải quyết được cả hai vấn đề nếu trên: dịch vụ cung cấp cho mỗi doanh nghiệp sử dụng nên giá dịch vụ sẽ rẻ hơn so với tự phát triển; bên thứ ba được đảm bảo và chứng nhận độc lập với CSP hoàn toàn đã tránh được các rủi ro chủ quan phát sinh. Nói tóm lại, sự xuất hiện và phát triển của một bên thứ ba đảm nhiệm vai trò bảo mật cơ sở dữ liệu trên dịch vụ đám mây xuất phát từ nhu cầu thực tế của các bên liên quan, hứa hẹn sẽ mang đến những tiện ích và trải nghiệm tuyệt vời cho người dùng, nhất là các doanh nghiệp và là hướng đi tiềm năng phát triển trong tương lai.

1.2. Báo cáo vấn đề

Trước sự phát triển mạnh mẽ của các dịch vụ lưu trữ đám mây, nguy cơ tấn công vào các cơ sở dữ liệu này đang tiềm ẩn trên hầu hết khu vực và đa dạng các đối tượng nạn nhân, kể cả các công ty lớn chuyên cung cấp dịch vụ lưu trữ đám mây như Google Cloud hay AWS.

McAfee đã tiến hành một nghiên cứu về các cuộc tấn công mạc vào các dịch vụ đám mây để xác định xem liệu có sự gia tăng các cuộc tấn công kể từ khi đại dịch Covid-19 bắt đầu hay không. Kết quả cho thấy sự gia tăng lên đến 630% các cuộc tấn công mạng vào các dịch vụ đám mây kể từ tháng 1 đến tháng 4 năm 2020. Với sự mở rộng của các dịch vụ chăm sóc sức khỏe qua điện thoại, nhiều nhà cung cấp dịch vụ đã chuyển sang sử dụng dịch vụ đám mây. Trong quý đầu tiên năm 2020, đây là ngành bị nhắm mục tiêu đa số trong các cuộc tấn công với khoảng 198 triệu IP độc hại được phát hiện [5](xem Hình 1.1).

Nhận thấy tình trạng như vậy, các nghiên cứu nỗ lực đảm bảo an toàn cho dữ liệu đang được thúc đẩy nhanh chóng, với mong muốn có thể đảm bảo tính bí mật cho dữ liệu, kể cả trong trường hợp bị mất mát, các phương pháp mã hóa – giải mã và kiểm soát truy cập đang được phát triển nhằm ứng dụng lên cơ sở dữ liệu quan hệ với mong muốn tăng tính bảo mật. Đồng thời, các giải pháp trên cũng phải có chi phí phù hợp với khả năng của doanh nghiệp, cá nhân. Quản lý truy cập dựa trên thuộc tính và mã hóa hỗ trợ thao tác được mong đợi sẽ góp phần tạo nên những bước tiến trong bảo mật dữ liệu, giảm đi đáng kể thiệt hại từ các cuộc tấn công, lỗi của người dùng hoặc CSP gây ra.



Hình 1.1: Số tổ chức bị tấn công[5]

1.3. Cơ sở lý thuyết và phương pháp luận

Trong khóa luận này, nhóm tác giả sử dụng kỹ thuật kiểm soát truy cập dựa trên thuộc tính, thông qua các phiên bản ứng dụng của mã hóa dựa trên thuộc tính. Đây là kỹ thuật được phát triển thêm từ kiểm soát truy cập dựa trên vai trò. Kiểm soát truy cập dựa trên thuộc tính không phân chia vai trò cho từng người dùng mà cấp quyền truy cập cho họ thông qua một tập thuộc tính biểu thị chính sách truy cập. Người dùng có những thuộc tính thỏa mãn chính sách được quy định có thể đọc được tài liệu mã hóa.

Bên cạnh đó tìm hiểu, nghiên cứu một số phương pháp thao tác trên cơ sở dữ liệu mã hóa để đảm bảo quyền riêng tư dữ liệu của người dùng trước các máy chủ đám mây bên thứ ba không đáng tin cậy.

1.4. Đối tượng và phạm vi thực hiện

Đối tượng nghiên cứu:

- Kỹ thuật kiểm soát truy cập dựa trên thuộc tính, kỹ thuật mã hóa dựa trên thuộc tính, phương pháp mã hóa chính sách khóa dựa trên thuộc tính, phương pháp mã hóa chính sách bản mã dựa trên thuộc tính.
- Mã hóa đồng cấu, lược đồ searchable encryption.
- Hệ quản trị cơ sở dữ liệu quan hệ có cấu trúc SQL.

Phạm vi thực hiện: Phân tích và đánh giá mô hình kiểm soát truy cập dựa trên thuộc tính trên dữ liệu mã hóa; hỗ trợ các thao tác tìm kiếm, cập nhật trên dữ liệu mã hóa; kết hợp triển khai trên hệ quản trị cơ sở dữ liệu MySQL.

Mục tiêu nghiên cứu: Tìm hiểu về các kỹ thuật, phương pháp, thuật toán nhằm kiểm soát truy cập dựa trên thuộc tính; hỗ trợ các thao tác trên dữ liệu mã hóa; từ đó xây dựng một ứng dụng nhằm đánh giá mức độ hiệu quả của phương pháp đối với khả năng bảo mật dữ liệu trên dịch vụ đám mây.

1.5. Cấu trúc luận văn

Luận văn này được chia thành 5 phần. Chương 1 giới thiệu những nét khái quát chung và những vấn đề cơ bản đặt ra khiến nghiên cứu này là cần thiết. Chương 2 và chương 3 trình bày một số nghiên cứu hiện tại về kiểm soát truy cập và mã hóa dữ liệu, cũng như đưa ra một số chi tiết về phương pháp của nhóm tác giả cho vấn đề được nêu ra ở chương 1. Chương 4 trình bày mô tả ứng dụng hiện thực về kiểm soát truy cập dựa trên mã hóa CP-ABE. Chương 5 tổng kết lại một số kết quả đạt được và hướng phát triển của nghiên cứu.

CHƯƠNG 2. **ĐIỀU KHIỂN TRUY CẬP TRÊN DỮ LIỆU MÃ HÓA**

2.1. Tổng quan về kiểm soát truy cập trên cơ sở dữ liệu

Kiểm soát truy cập là vấn đề quan trọng trong bất kỳ cơ quan, tổ chức hay hệ thống nào. Từ thời cổ đại, con người đã phát minh ra các phương pháp kiểm soát truy cập khác nhau: sử dụng lệnh bài, giấy tờ hợp lệ… để ra vào các khu vực giới hạn. Trong lĩnh vực công nghệ thông tin hiện đại, kiểm soát truy cập là khả năng của hệ thống để xác định xem người dùng có thể truy cập vào một phần dữ liệu cụ thể được lưu giữ trong hệ thống máy tính và môi trường hoạt động liên quan của nó hay không [6]. Nó bao gồm hai thành phần chính là xác thực *(authentication)* và ủy quyền *(authorization).* Xác thực là phương pháp xác minh danh tính của người đang truy cập cơ sở dữ liệu. Ủy quyền là phương pháp xác định người dùng có được phép truy cập vào dữ liệu hoặc thực hiện các thao tác trên dữ liệu đó hay không. Nếu thiếu một trong hai yếu tố này, dữ liệu không được bảo vệ.

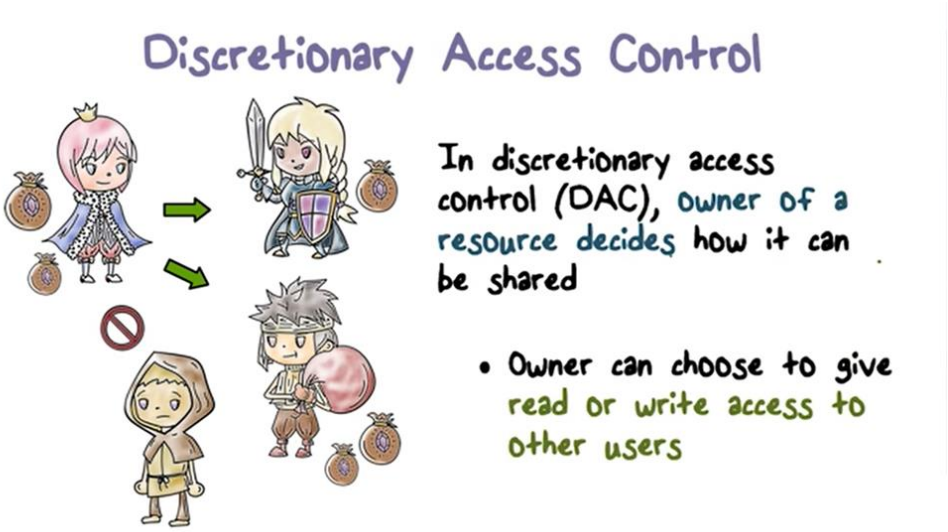
Các chính sách kiểm soát truy cập có thể chia thành ba nhóm chính chính: Điều khiển truy cập tùy ý *(DAC)*, Điều khiển truy cập bắt buộc *(MAC)* và Điều khiển truy cập theo vai trò *(RBAC)* [7].

2.1.1. Điều khiển truy cập tùy ý (DAC).

Ở mô hình điều khiển truy cập DAC, chủ sở hữu kiểm soát quyền truy cập nhưng chỉ với bản gốc, không phải với bản sao. Mô hình gồm ba thành phần: chủ thể *(subject),* quyền truy cập *(access)* và đối tượng *(object).* Trong DAC, chủ sở hữu dữ liệu quyết định quyền truy cập của các chủ thể khác trên đối tượng do mình sở hữu (xem Hình 2.1).

Tuy nhiên, mô hình này có nhược điểm lớn là việc trao quyền kiểm soát truy cập của đối tượng chủ sở hữu của nó có thể làm tiết lộ dữ liệu cho những người không có quyền truy cập hợp pháp với nó. Chúng ta không thể đảm bảo rằng chủ sở hữu dữ liệu là tuyệt đối tin tưởng được. Trong trường hợp chủ sở hữu có thể tin tưởng, khả năng chủ sở hữu quyết định sai các quyền truy cập trên các tài nguyên của mình có thể gây nên đổ vỡ hệ thống. Các chủ thể xấu tồn tại trong hệ thống có thể lợi dụng sự thiếu hiểu biết

hoặc sai lầm của chủ sở hữu dữ liệu, kết hợp với nhau để chiếm đoạt tài nguyên bất hợp pháp.



Hình 2.1: Minh họa mô hình DAC[1]

Giả sử hệ thống chúng ta có các thành phần như mô tả trong bảng sau:

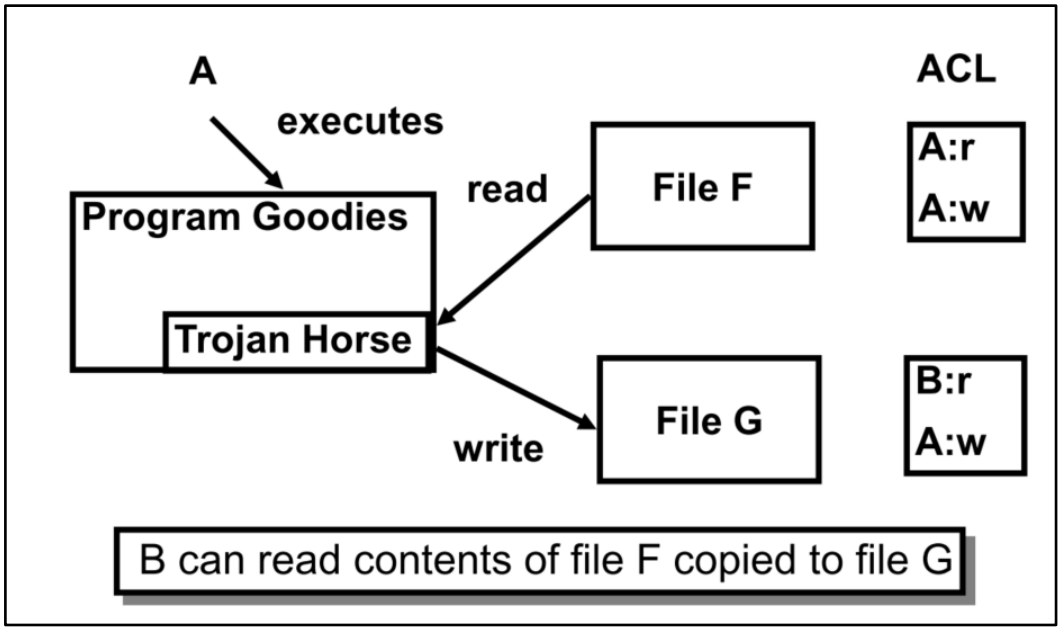
Chủ thể	Quyền truy cập	Đối tượng
A	R	File F
A	W	File F
A	W	File G
B	R	File G

Bảng 2.1: Ví dụ về kiểm soát truy cập DAC

Theo như bảng trên, B không thể đọc được các nội dung trong file F do không có quyền R trên file F. Tuy nhiên, tận dụng lỗ hổng của mô hình DAC, chủ thể A có thể đọc các nội dung trên file F, thực hiện ghi nó vào file G. Từ đó, B có thể đọc được nội dung của file F mà không cần có quyền gì trên file F (xem Hình 2.2).

[1] Nguồn hình ảnh: youtube.com/watch?v=UNRnSaXajC4

Vì những rủi ro của nó, trong các hệ thống hiện đại ngày nay, các công ty, tổ chức hạn chế và hầu như không sử dụng mô hình kiểm soát truy cập DAC.

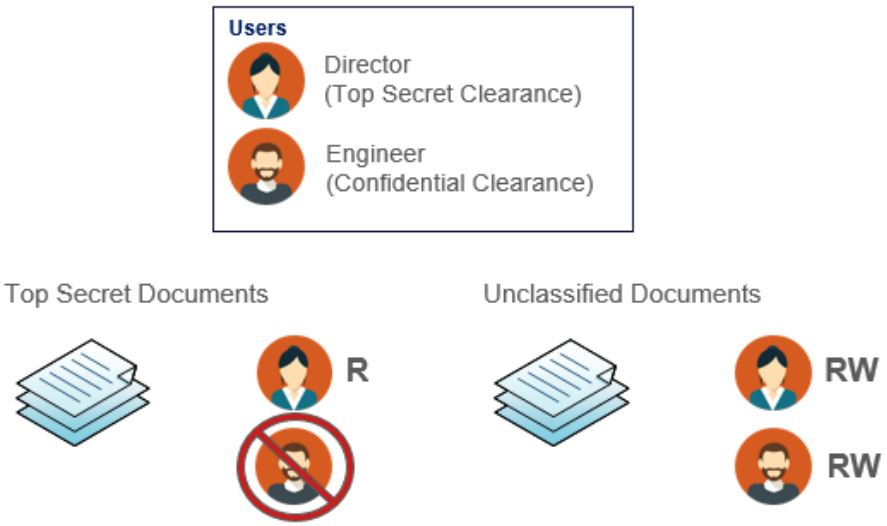


Hình 2.2: Lỗ hổng trojan horse trong mô hình DAC[7]

2.1.2. Điều khiển truy cập bắt buộc *(MAC)*.

Trong mô hình MAC, quyền truy cập đối với các đối tượng không được quyết định bởi chủ thể mà được quyết định bởi một người *(hoặc hệ thống)* quản trị. Các chủ thể và đối tượng được gắn các nhãn tương ứng với mức độ *"nhạy cảm"* của họ và được sắp xếp một cách có trật tự (xem hình Hình 2.3). MAC áp dụng hai quy tắc chính đó là ***no read-up*** và ***no write-down***. Mối quan hệ giữa các luồng thông tin trong hệ thống này là phản xạ, bắc cầu và phản đối xứng. Điểm mạnh của MAC là đảm bảo bí mật của thông tin và cung cấp biện pháp chống lại phương pháp tấn công trojan horse. Điều này đạt được do các nhãn của đối tượng được truyền qua các bản sao và ngăn chặn việc hạ cấp nhãn của đối tượng, từ đó đảm bảo dữ liệu không thể chuyển từ mức độ *"nhạy cảm"* cao xuống thấp [8].

Khi nhắc tới MAC, mô hình Bell-La Padula [9] thường được nhắc tới thường xuyên nhất. Nhóm tác giả phân tích rất khái quát về mô hình Bell-La Padula nhằm đại diện cho phương pháp MAC.



Hình 2.3: Minh họa mô hình MAC[2]

Trong mô hình Bell-La Padula, có một tập hợp nhãn phân loại *(top-secret, secret, confidential, unclassified)* được xác định hoàn toàn. Bên cạnh đó, có một danh sách các danh mục không được phân loại. Sự kết hợp của một nhãn và một thành phần con của danh mục được gọi là mức bảo mật. Mức bảo mật được sắp xếp một phần và tạo thành một mạng lưới [10].

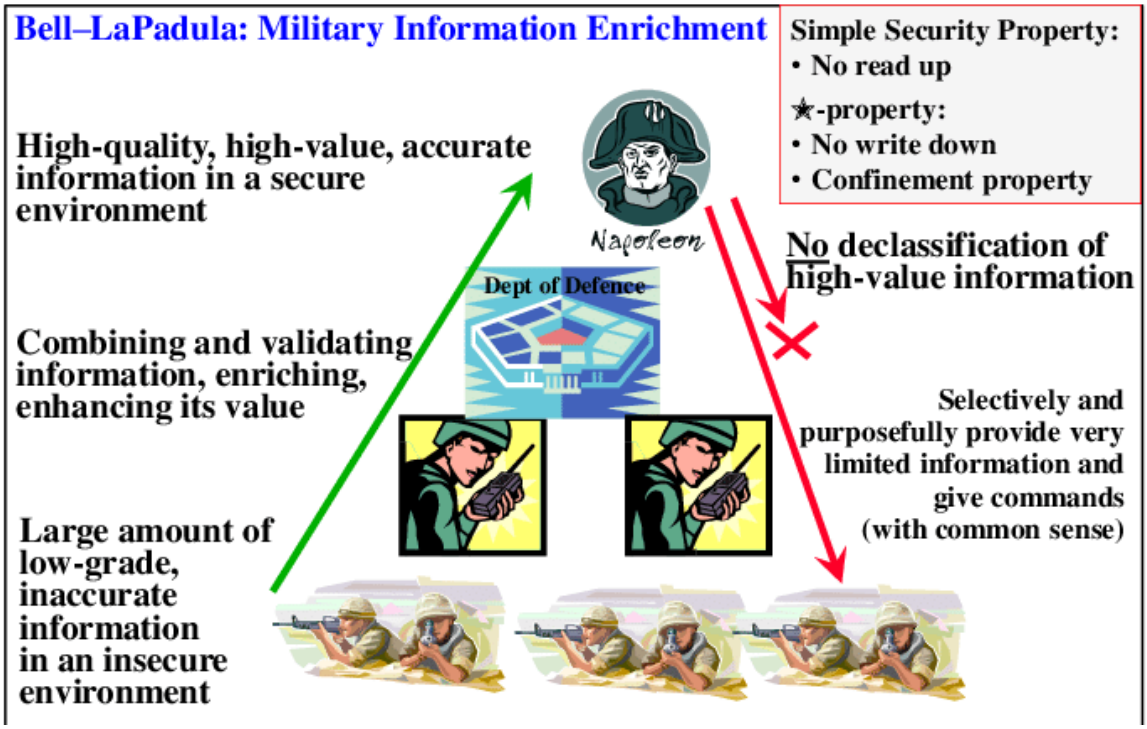
Trong mạng lưới này, các chủ thể không được đọc các tài liệu có nhãn mức bảo mật cao hơn mình, đồng thời không được viết ra các tài liệu có nhãn bảo mật thấp hơn mình. Nghĩa là một chủ thể có mức bảo mật *secret* thì không thể đọc được tài liệu có nhãn *top-secret* nhưng có thể đọc được các tài liệu có nhãn bằng hoặc thấp hơn mình *(secret, confidential, unclassified).* Bên cạnh đó, chủ thể này cũng không thể tạo ra các tài liệu có nhãn bảo mật thấp hơn nhãn của mình. Phương pháp kiểm soát truy cập này

[2] Nguồn hình ảnh: https://community.aras.com/b/english/posts/access-control-methods-in-innovator

rất phù hợp trong môi trường cần đề cao tính bí mật của thông tin như quân sự, ngoại giao (Hình 2.4).

Tuy nhiên, do sự đa dạng các yêu cầu, đặc biệt là mong môi trường điện toán đám mây với lượng truy cập lớn, các quy tắc và định lý bảo mật hiện tại trong BLP không thể thích ứng được [11]. Các hạn chế của nó dần dần bộc lộ trong quá trình phát triển lý thuyết về bảo mật.

Giả sử, trong hệ thống ban đầu chỉ có một một loại phân cấp, khi một chủ thể yêu cầu truy cập vào đối tượng. Trong trường hợp này, do tất cả chủ thể và đối tượng đều được phân cấp là thấp nhất, yêu cầu này đương nhiên được chấp thuận theo BLP, nhưng đó rõ ràng là rủi ro lớn [12].



Hình 2.4: Ví dụ về mô hình Bell-La Padula[3]

[3] Nguồn hình ảnh: https://www.researchgate.net/figure/The-Bell-LaPadula-Model_fig3_300409231

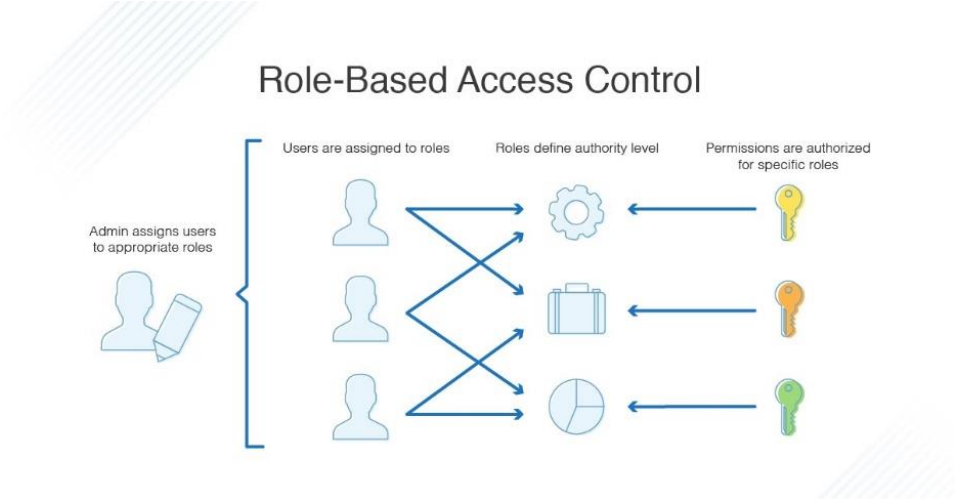
Nhiều giải pháp cải thiện mô hình BLP được đưa ra để phù hợp với sự phát triển của lý thuyết bảo mật: đề xuất một phương pháp có thể xác định các chủ thể một cách linh động bằng cách cung cấp các phương pháp chuyển đổi trạng thái trong BLP [13]; định nghĩa các chủ thể, đối tượng và quy tắc bảo mật trong network domain, cung cấp các quy tắc để hạn chế hành vi của hệ thống [14].

Mặc dù các nghiên cứu đã có nhiều nỗ lực để cải thiện BLP và các ứng dụng của nó, nhưng mô hình chủ động điều chỉnh với những thay đổi ít khi được đề cập đến. Các nghiên cứu ít chú đến các hệ thống yêu cầu thời gian thực, thiếu khả năng thay đổi theo bối cảnh [15].

2.1.3. Điều khiển truy cập theo vai trò *(RBAC)*

Phương pháp điều khiển truy cập dựa theo vai trò thực hiện dựa trên chính sách quy định quyền truy cập của người dùng vào các tài nguyên trên cơ sở mà người dùng thực hiện. Nó yêu cầu xác định cụ thể từng vai trò trong hệ thống. Theo đó, vai trò là một tập hợp các quyền sử dụng các tài nguyên phù hợp với chức năng và công việc của một người, là một tập hợp các hành động và trách nhiệm liên quan đến một hoạt động cụ thể. Thay vì chỉ định tất cả các quyền mà người dùng được phép thực thi, các quyền truy cập được chỉ định cho các vai trò. Người dùng được thêm vào các tập hợp vai trò mà thông qua đó, được phép thực thi các quyền của vai trò đó [16].

Theo thời gian, các doanh nghiệp nhận thấy nhu cầu vượt ra ngoài các định nghĩa về nhóm người dùng và quyền của RBAC. Chúng cần bao gồm các thuộc tính linh động như thời gian trong ngày, vị trí của người dùng. Các hệ thống phân tán và thay đổi liên tục, kiểm soát truy cập dựa trên thuộc tính là một lựa chọn hoặc thay thế, hoặc bổ trợ cho RBAC. ABAC sử dụng các đối tượng được gắn nhãn và thuộc tính người dùng thay vì quyền để kiểm soát một cách linh hoạt.



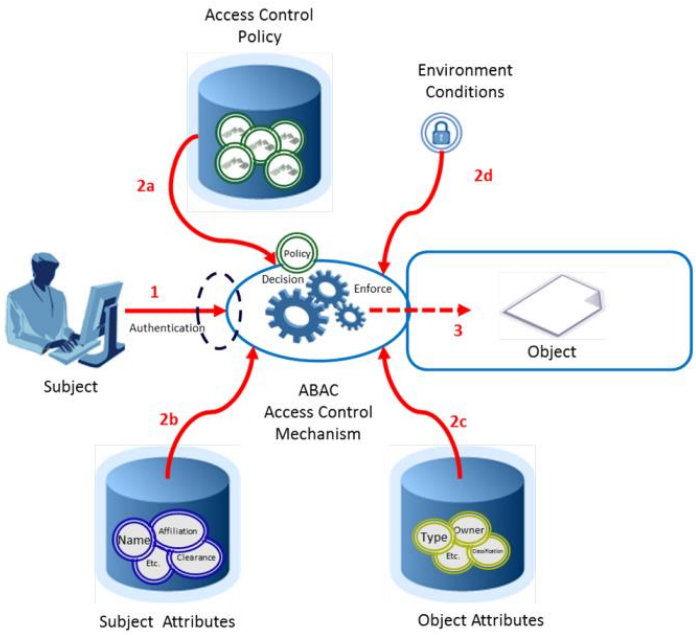
Hình 2.5: Minh họa về RBAC[4].

2.1.4. Điều khiển truy cập dựa theo thuộc tính (ABAC)

ABAC là một mô hình logic điều khiển truy cập vào các đối tượng bằng cách đánh giá các quy tắc dựa trên các thuộc tính của các đối tượng và chủ thể, các hoạt động và môi trường liên quan đến một ngữ cảnh yêu cầu cụ thể. Nó cho phép kiểm soát truy cập chính xác hơn bằng cách cho phép số lượng lớn đầu vào rời rạc tham gia vào quá trình quyết định quyền truy cập, và do đó cung cấp một lượng lớn hơn các kết hợp có thể có của các thuộc tính để phản anh một chính sách lớn hơn rất nhiều so với các phương pháp khác. Nó chỉ bị giới hạn ngôn bởi ngôn ngữ tính toán và sự phong phú của các thuộc tính có sẵn [17].

Mô hình điều khiển truy cập ABAC cơ bản có 3 bước: chủ thể yêu cầu một truy cập (đọc, ghi, thực thi…) tới một đối tượng cụ thể; hệ thống kiểm tra và đánh giá các thuộc tính mà chủ thể cung cấp; chủ thể được cho phép truy cập tài nguyên nếu là người dùng xác thực và bị cấm nếu không xác thực.

[4] https://www.dnsstuff.com/rbac-vs-abac-access-control



Hình 2.6: Mô hình ABAC cơ bản[1]

ABAC cho phép đưa ra các quyết định kiểm soát truy cập mà không cần chủ thể biết trước về đối tượng. Với phương pháp khái quát thuộc tính được xác định nhất quán giữa các tổ chức, nó tránh được nhu cầu xác minh quyền truy cập rõ ràng của các chủ thể với các đối tượng, nghĩa là chủ thể không biết rõ mình có quyền truy cập được đến đối tượng hay không khi chưa có nhu cầu truy cập tài nguyên đó, trong khi đối với RBAC, các chủ thể trong một tập vai trò biết rõ và tường minh các quyền của mình đối với đối tượng cụ thể. ABAC cho phép sự linh hoạt trong một doanh nghiệp lớn và rất lớn, nơi việc quản lý các danh sách vai trò và nhóm kiểm soát truy cập tốn thời gian và phức tạp.

Nếu các thuộc tính được xác định nhất quán, các thủ tục xác thực và cho phép truy cập có thể được thực thi và quản lý trong các cơ sở khác nhau, đồng thời duy trì mức độ bảo mật thích hợp trong tổ chức mới. Ví dụ một chủ thể có thể xác thực và truy cập tài nguyên trong bệnh viện này và sau đó được ủy quyền để truy cập các tài nguyên trên bệnh viện khác tương tự như đối với bệnh viện cũ, nhờ các giá trị thuộc tính được xác định nhất quán từ trước. Sau khi hoàn thành vai trò tại cơ sở tạm thời, việc loại bỏ

quyền được cấp tạm thời được thực hiện dễ dàng mà không ảnh hưởng tới những người dùng khác trong hệ thống.

Kỹ thuật kiểm soát truy cập quản lý dựa trên từng giá trị thuộc tính do người sở hữu dữ liệu định nghĩa. Nó hỗ trợ quản lý các nhóm người dùng có cùng chính sách truy cập như kỹ thuật kiểm soát dựa trên vai trò, nhưng cá nhân hóa hơn nhờ áp dụng phương pháp dẫn xuất khóa nhóm. Người quản trị không cần phải thay đổi nhóm vai trò của từng người dùng mỗi khi họ thay đổi nhiệm vụ hoặc rời khỏi hệ thống mà việc phân bố lại nhóm trong hệ thống được thực hiện một cách tự động hoàn toàn. Khi mỗi người dùng có thay đổi về thuộc tính, gia nhập hoặc rời khỏi hệ thống, người quản trị chỉnh sửa tương ứng giá trị thuộc tính của họ, những người dùng khác trong hệ thống không bị ảnh hưởng. Thông thường, chính sách truy cập quy định một chủ thể **s** có thể truy cập vào tài nguyên **r** trong điều kiện bảo mật **e** được biểu diễn bằng biểu thức đại số bởi 3 biến **s**, **r**, **e** như sau:

$$can\_access(s, r, e) \leftarrow f(ATTR_s, ATTR_r, ATTR_e)$$ [18]

với $ATTR_s$ là thuộc tính của chủ thể s.

Thông thường, một người dùng sẽ được chỉ định (hủy chỉ định) một cách tự động vào các nhóm nếu họ đáp ứng các điều kiện thành viên nhóm. Một điều quan trọng khác là cơ chế quản lý khóa nhóm, bởi mục tiêu các nhóm thường là chia sẻ dữ liệu. Do đó dữ liệu phải được mã hóa với các khóa chỉ được cung cấp cho các thành viên của nhóm. Việc quản lý những khóa, bao gồm các lựa chọn, phân phối, lưu trữ, cập nhật yêu cầu phương pháp quản lý khóa nhóm dựa trên thuộc tính. Theo đó các khóa nhóm được gán cho người dùng dựa trên các thuộc tính nhận dạng của họ.

Khi nhóm thay đổi, khóa nhóm mới phải được chia sẻ với các thành viên hiện có, để một thành viên nhóm mới không thể truy cập dữ liệu được truyền trước khi họ tham gia nhóm và một người dùng đã rời đi khỏi nhóm không thể truy cập dữ liệu của nhóm. Một vấn đề khác là bảo vệ chống lại các cuộc tấn công thông đồng mà qua đó, một nhóm

người dùng gian lận thông đồng có thể có được khóa nhóm bằng cách tập hợp những khóa của nhau, sau đó lấy khóa nhóm và đáng ra họ không được phép lấy riêng lẻ.

Mặc dù ABAC có thể hỗ trợ trong việc chia sẻ thông tin doanh nghiệp, nhưng khi triển khai trên quy mô doanh nghiệp, tập hợp các khả năng cần thiết để thực hiện trở nên phức tạp hơn. Ở cấp độ doanh nghiệp, quy mô lớn đòi hỏi khả năng quản lý phức tạp và đôi khi cần được thiết lập một cách độc lập.

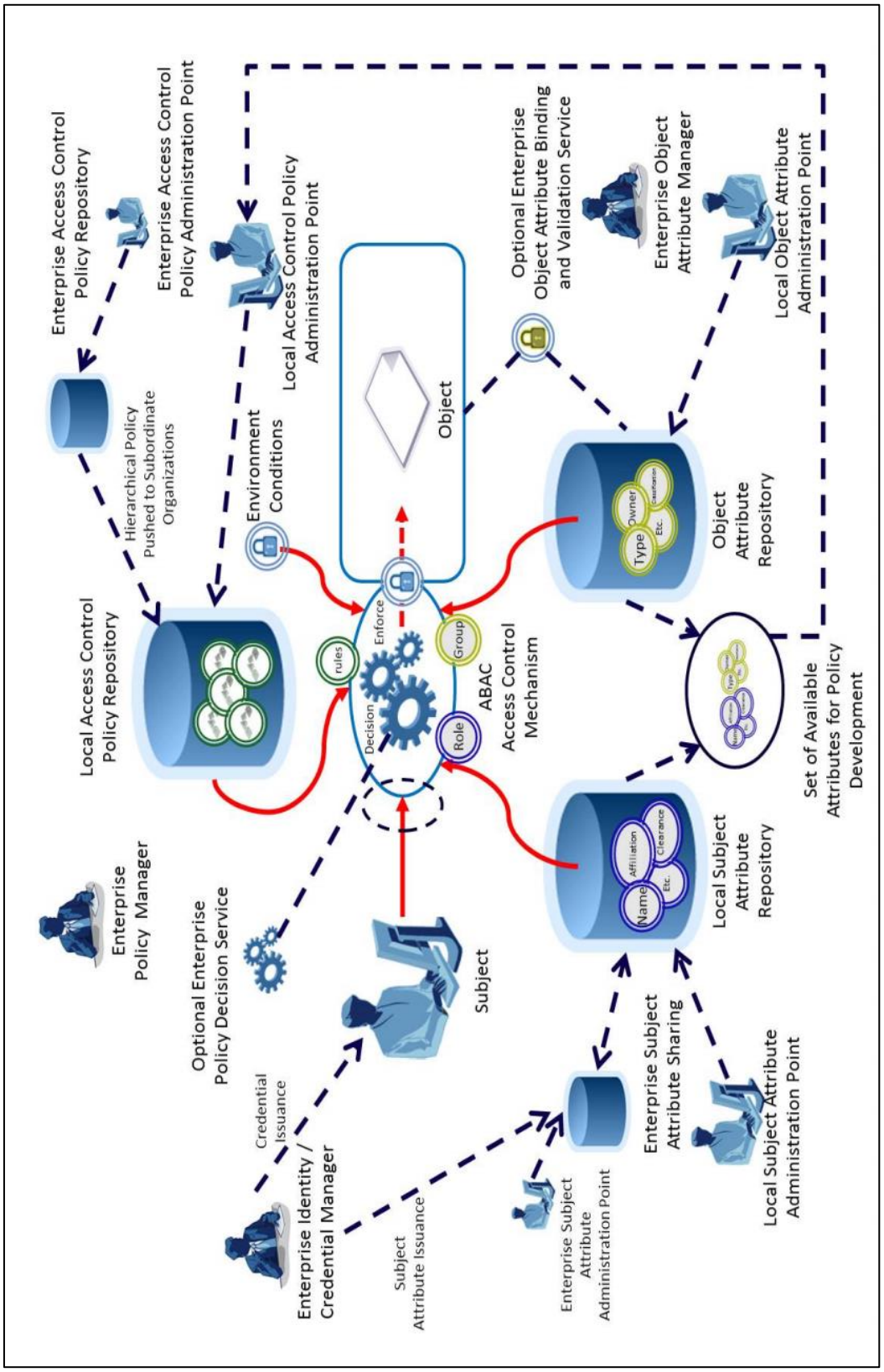
Hình 2.7 trình bày chi tiết các thành phần cần thiết khi triển khai ABAC trong doanh nghiệp. Hầu hết các doanh nghiệp hiện tại đều có thể đáp ứng được các thành phần trong sơ đồ như có hình thức quản lý và thông tin xác thực nhân viên. Tuy nhiên, các quy tắc thường được viết thành các loại văn bản mà con người có thể đọc được và thiết kế thành các phần mềm riêng lẻ. Để có thể triển khai ABAC trong doanh nghiệp được hiệu quả, các chính sách quản lý phải lưu dưới dạng máy có thể đọc được. Từ đó có thể triển khai quản lý các chủ thể và đối tượng được toàn diện.

Trong ABAC, chúng ta không cần định nghĩa các vai trò cụ thể và cấp quyền truy cập cho các vai trò đó như với RBAC. Trong ví dụ tiếp theo, một hệ thống quản lý xếp loại phim và độ tuổi người dùng được phép xem phim hàng tháng. Xếp loại phim và điều kiện xem phim quy định trong Bảng 2.2

Loại phim	**Người dùng cho phép**
R	Tuổi 21 hoặc lớn hơn
PG-13	Tuổi 13 hoặc lớn hơn
G	Tất cả

Bảng 2.2: Ví dụ về loại hình ABAC

Như vậy, chính sách quản lý truy cập trong hệ thống này quy định chỉ người lớn mới được xem loại phim R, trẻ vị thành niên hoặc lớn hơn mới được được loại phim PG-13 và phim G thì dành cho tất cả các đối tượng.



Hình 2.7 Ví dụ về ABAC trong doanh nghiệp [1]

Đối với ABAC, hệ thống không quy định các vai trò người lớn, trẻ vị thành niên hay trẻ em em mà dựa vào các thuộc tính vốn có (ở ví dụ này là tuổi) để định nghĩa chính sách truy cập trong hệ thống. Một người sử dụng hệ thống u có thể truy cập vào và xem bộ phim m (trong điều kiện e mà ở đây không nhắc tới) phải thỏa mãn chính sách truy cập:

R1: $can\_access(u, m, e) \leftarrow (Age(u) \geq 21 \wedge Rating(m) \in (R, PG\_13, G)) \vee (Age(u) \geq 13 \wedge Rating(m) \in (PG\_13, G)) \vee (Age(u) < 13 \wedge Rating(m) \in G)$

Các chính sách truy cập chi tiết hơn thường liên quan đến nhiều thuộc tính của các chủ thể và đối tượng. Trong các trường hợp như vậy, ABAC dễ dàng quản lý và mở rộng hơn. Để minh họa điều này, nhóm tác giả mở rộng ví dụ một chút: giả sử phim được phân loại thêm về chất lượng phim được xem. Người dùng đã trả phí (premium) được xem loại phim có độ phân giải full-HD trong khi người dùng thường (nomal) chỉ được xem loại phim có độ phân giải HD. Trong ví dụ này, chính sách quy định R1 vẫn được áp dụng và bên cạnh đó, áp dụng thêm:

R2: *can_access*(u, m, e) ← (MembershipType(u) = 'Premium') ∨ (MembershipType(u) = 'Normal' ∧ MovieType(m) = 'HD')

và chính sách truy cập cuối cùng sẽ là:

R3: *can_access*(u, m, e) ← ***R1*** ∧ ***R2***

Bên cạnh đó, chính sách đối với điều kiện **e** cũng có thể áp dụng tương tự, như trẻ em chỉ được xem phim trong khung giờ cho phép hoặc các loại phim có yếu tố chính trị, tôn giáo có thể xem xét địa điểm truy cập của người dùng.

2.2. Quản lý truy cập dựa trên thuộc tính.

Như đã đề cập ở phần trên, mô hình quản lý truy cập dựa trên thuộc tính có ưu điểm lớn so với các mô hình cũ. Hầu hết các doanh nghiệp ngày nay đang áp dụng các giải pháp quản lý định danh nhân viên, các thông tin về bộ phận làm việc, cấp bậc, chuyên môn… đều được lưu trữ và quản lý tập trung trên các phần mềm nhân sự (hoặc

các phần mềm tùy biến do doanh nghiệp thiết kế). Điều quan trọng là làm sao tận dụng được các giải pháp này để quản lý các nhóm nhằm hiện thực ABAC.

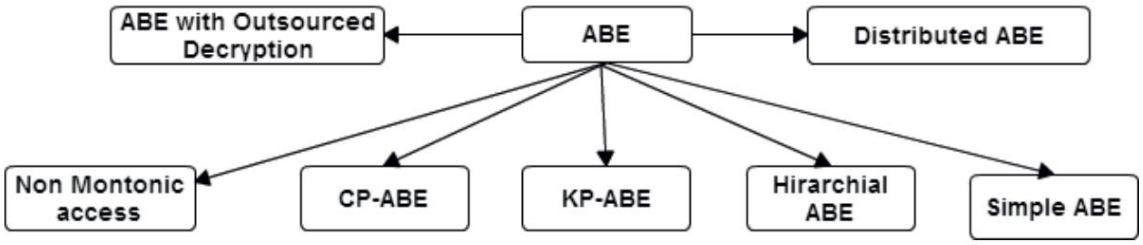
Một yêu cầu quan trọng khác là cung cấp cơ chế quản lý khóa nhóm, vì mục tiêu của một nhóm là thường xuyên chia sẻ dữ liệu. Dữ liệu phải được mã hóa bằng các khóa chỉ thành viên trong nhóm đó biết. Việc quản lý các khóa này, bao gồm thêm mới, phân phối, lưu trữ và cập nhật phải hỗ trợ hiệu quả cho mô hình ABAC.

Không giống như việc quản lý theo mô hình RBAC, nơi mà hệ thống cấp quyền cho các vai trò (role), sau đó quản lý người dùng trong các vai trò, ABAC cung cấp quyền trực tiếp cho người dùng thông qua các thuộc tính mà người đó sở hữu. Thách thức lớn là phương pháp quản lý khóa hiệu quả dành cho nhóm người dùng có chung một số thuộc tính. Khi nhóm này thay đổi (thêm hoặc bớt người dùng) thì họ có/không truy cập hợp lệ tới các tài nguyên của nhóm, trong khi các thành viên cũ của nhóm không bị ảnh hưởng. Quá trình cấp khóa mới cho nhóm phải đúng với chính sách truy cập, đồng thời cũng không tác động lớn tới các thành viên cũ. Bên cạnh đó, việc chống lại các cuộc tấn công thông đồng của những người dùng xấu cũng đặt ra một thử thách.

Trong ví dụ ở phần 2.2, giả sử mỗi thành viên của hệ thống được cấp một khóa để có thể truy xuất vào bộ phim mà họ muốn xem. Khi một người dùng ‘Premium’ không tiếp tục trả phí, hệ thống cần loại bỏ quyền của họ đối với chất lượng full-HD của bộ phim M mà không cần tác động đến khóa của họ (thực tế là không thể tác động vì phải giao khóa cho người dùng), trong khi những người dùng khác đã có khóa được cấp hợp lệ vẫn xem được như cũ mà không cần cấp lại toàn bộ khóa mới. Việc chống lại các cuộc tấn công phối hợp như một người dùng nhỏ hơn 21 tuổi nhưng có hạng ‘Premium’ và một người dùng lớn hơn 21 tuổi có hạng “Normal’ có thể phối hợp với nhau để xem bộ phim xếp loại R với chất lượng full-HD.

Nhằm hiện thực quản lý truy cập dựa trên thuộc tính, đồng thời giữ được tính bí mật của dữ liệu được lưu trên các dịch vụ đám mây, phương pháp mã hóa dựa trên thuộc

tính, Attribute-based Encryption được phát triển với nhiều biến thể khác nhau được phân loại như Hình 2.8.



Hình 2.8: Phân loại ABE [19]

Trong phần tiếp theo, nhóm tác giả trình bày chi tiết thuật toán của hai phương pháp mã hóa dựa trên thuộc tính trong Hình 2.8 là CP-ABE và KP-ABE. Đây là hai phương pháp mã hóa nổi bật nhất. Các nghiên cứu về ABE theo hai hướng này ngày càng rộng và tăng nhanh.

Cả KP-ABE và CP-ABE đều sử dụng chung một cấu trúc truy cập, được gọi là cây truy cập. Gọi $\mathcal{T}$ là một cây biểu thị cấu trúc truy cập. Mỗi điểm nút trong cây là một cổng truy cập. Một cổng có cấu trúc bao gồm các nút con của nó và một giá trị cổng. Giả sử $\boldsymbol{n_x}$ là số lượng nút con của nút $\boldsymbol{x}$ và $\boldsymbol{t_x}$ là giá trị cổng thì ta có $0 < t_x \leq n_x$. Nếu $t_x = 1$ thì cổng đó là cổng OR, và ngược lại, nếu $t_x = n_x$ thì đó là cổng AND. Mỗi nút lá (là nút không có nút nào là nút con của nó) diễn giải một thuộc tính và có giá trị cổng $t_x = 1$. Hàm $\boldsymbol{att(x)}$ lúc này trả về thuộc tính liên kết với nút lá x. Mỗi nút trong $\mathcal{T}$ được đánh số thứ tự duy nhất. Hàm $\boldsymbol{index(x)}$ trả về số thứ tự của nó, hàm $\boldsymbol{parent(x)}$ trả về nút cha của nó.

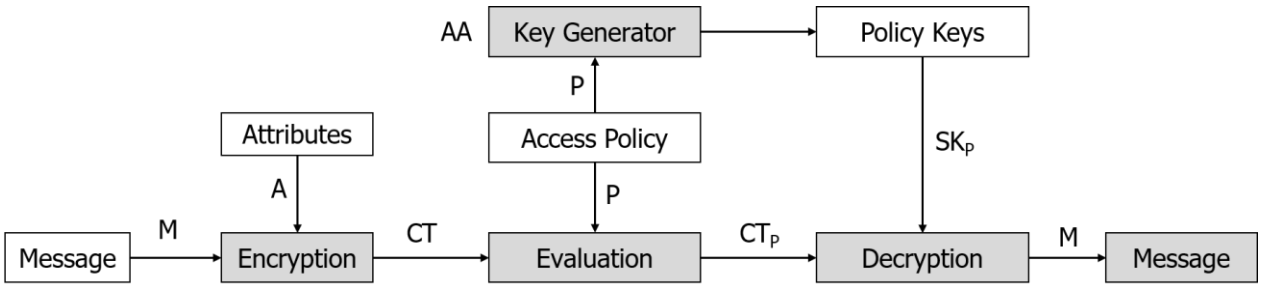
Giả sử $\mathcal{T}$ là cây truy cập với nút gốc tại r, $\mathcal{T}_x$ là cây con của $\mathcal{T}$ có nút gốc tại x. Tập thuộc tính λ được gọi là thỏa mãn $\mathcal{T}_x$ nếu $\mathcal{T}_x(\lambda) = 1$. Việc tính toán $\mathcal{T}_x(\lambda)$ được thực hiện:

- Nếu x là nút lá, $\mathcal{T}_x(\lambda) = 1 \Leftrightarrow att(x) \in \lambda$.

- Nếu x không phải là nút lá, gọi các nút con của x là x'. Lần lượt tính toán các cây con $\mathcal{T}_{x'}(\lambda)$. Nếu có ít nhất t_x cây con $\mathcal{T}_{x'}(\lambda) = 1$ thì $\mathcal{T}_x(\lambda) = 1$. Ngược lại, tập λ không thỏa mãn cấu trúc truy cập được quy định trong $\mathcal{T}_x$.

2.2.1. Mã hóa dựa trên thuộc tính sử dụng chính sách khóa

Phần tiếp theo, nhóm tác giả trình bày phương pháp mã hóa dựa trên phương pháp mã hóa khóa nhóm [20]. Hình 2.9 mô tả quá trình thực hiện các bước của phương pháp KP-ABE (Key-Policy Attribute-based Encryption). Dữ liệu M được mã hóa thành bản mã CT theo tập thuộc tính A, được kiểm soát qua chính sách truy cập P để tạo ra bản mã mới dựa trên P (CT_P) mà chỉ có thể giải mã bằng khóa bí mật SK_P được cấp từ chính sách P. Sơ đồ yêu cầu bên thứ ba AA đáng tin cậy để tạo khóa phục vụ cho giải mã. Chi tiết các bước thực hiện mô tả ở phần tiếp theo.



Hình 2.9: Sơ đồ khối KP-ABE [21]

a. Thiết lập

Hàm **Setup(ℓ)** lấy tham số bảo mật ℓ là đầu vào. Hàm khởi tạo các giá trị số nguyên tố q có ℓ bit, số lượng tối đa nhóm người dùng trong hệ thống $N \geq n$, một hàm băm H, tập khóa KS, tập bí mật SS, tập các thông tin bí mật đã được sử dụng **S**.

Hệ thống định nghĩa một hệ số Lagrange $\Delta_{i,\mathbf{Q}}(x) = \prod_{j \in \mathbf{Q}, j \neq i} \frac{x-j}{i-j}$ với $i \in \mathbb{F}_q$ và tập **Q** bao gồm các phần tử trong $\mathbb{F}_q$.

Thuật toán 1: **Setup(ℓ)**

AA:

1: $q \leftarrow \{0,1\}^{\ell}$

2: $N \geq n$

3: $H(.) : \{0,1\}^{*} \rightarrow \mathbb{Z}_q$

4: $KS = \mathbb{Z}_q$

5: $SS \leftarrow \{0,1\}^{\ell}$

6: $\mathbf{S} \leftarrow$ empty set

7: $\Delta_{i,\mathbf{Q}}(x) = \prod_{j \in \mathbf{Q}, j \neq i} \frac{x-j}{i-j} \leftarrow i \in \mathbb{Z}_q, \mathbf{Q} \subseteq \mathbb{Z}_q$

Hình 2.10: Mã giả thuật toán Setup

b. Khởi tạo các chuỗi bí mật

Ở phần này, hệ thống thực hiện khởi tạo các giá trị bí mật cho mỗi người dùng trong hệ thống. Đây là giá trị mà với nó, hệ thống xác định một người dùng có truy cập hợp lệ với những thuộc tính mà họ có hay không.

Thuật toán 2: **SecGen($\boldsymbol{UA}, \boldsymbol{A}$)**

AA

REPEAT

1: $i \leftarrow (1, n)$

2: $\boldsymbol{S}_i \leftarrow \boldsymbol{UA}$

REPEAT

3: $att_j \leftarrow \boldsymbol{S}_i$

4: **IF** $(att_j \in \boldsymbol{A})$

5: $s_{i,j} \leftarrow SS$

6: $\beta_i += \langle i, s_{i,j} \rangle; \mathbf{S} += \langle i, s_{i,j} \rangle$

RETURN

7: $\boldsymbol{\beta} = \{\beta_i \mid 1 \leq i \leq n\}$

Hình 2.11: Mã giả thuật toán SecGen

Hàm **SecGen($\boldsymbol{UA}, \boldsymbol{A}$)** lấy đầu vào là tập tất cả thuộc tính sử dụng trong hệ thống $\boldsymbol{A}$ và ma trận các thuộc tính của người dùng $\boldsymbol{UA}$ và trả về tập $\boldsymbol{\beta} = \{\beta_i \mid 1 \leq i \leq n\}$. Gọi $\boldsymbol{S}_i \subset \boldsymbol{A}$ là tập thuộc tính của người dùng i. Đối mỗi mỗi giá trị thuộc tính $att_j \in \boldsymbol{S}_i$, hệ

thống chọn các chuỗi bí mật ngẫu nhiên $s_{i,j} \in SS$ duy nhất cho mỗi người dùng, thêm vào tập β_i và **S** giá trị $\langle i, s_{i,j} \rangle$

Người sử dụng hệ thống cung cấp tập thuộc tính $\boldsymbol{S_i}$ hợp lệ cũng được cấp các giá trị bí mật với các bước như trên.

c. Khởi tạo khóa

Thuật toán 3: **intermediateKey($\boldsymbol{S_i}$)**

REPEAT

1: $att_j \leftarrow \boldsymbol{S_i}$; $\mathbf{K}_i \leftarrow empty\ set$

2: $q_r(0) = k; d_r = t_r - 1 | k \leftarrow KS$

3: $\mathbf{R_r} = \{(x_t; y_t) | (x_t, y_t \in \mathbb{Z}_q; 1 \leq t \leq d_r\}$

4: $q_r = \sum_{i=1}^{t_r} q_r(x_i) . \prod_{\substack{j=1 \\ j \neq i}}^{t_r} \frac{x - x_j}{x_i - x_j} | \{q_r(x_i) = y_i; (x_i, y_i) \in \mathbf{R}\}$

5: **IF** $(x\ != leaf)$

6: $q_x(0) = q_{parent(x)}(index(x)); d_x = t_x - 1$

7: $\mathbf{R_x} = \{(x_t; y_t) | (x_t, y_t \in \mathbb{Z}_q; 1 \leq t \leq d_x\}$

8: $q_x = \sum_{i=1}^{t_x} q_x(x_i) . \prod_{\substack{j=1 \\ j \neq i}}^{t_x} \frac{x - x_j}{x_i - x_j} | \{q_x(x_i) = y_i; (x_i, y_i) \in \mathbf{R}\}$

9: **ELSE**

10: $q_x(0) = q_{parent(x)}(1) = k_{i,j}$

11: $\mathbf{K}_i += k_{i,j}$

12: **RETURN:** $\mathbf{K}_i$

Hình 2.12: Mã giả intermediateKey(S_i)

Ở hàm này, AA tạo khóa công khai cho các nhóm thuộc tính, đồng thời cung cấp khóa trung gian cho mỗi người dùng để họ có thể thực thi quyền truy cập thông qua các thuộc tính mà họ sở hữu. Hàm trả về giá trị công khai mà người dùng có thể dùng nó để dẫn xuất được khóa nhóm nếu thỏa mãn cây truy cập $\mathcal{T}$ được xây dựng cho chính sách truy cập **P.**

Gọi $\mathcal{T}_r$ là cây con của $\mathcal{T}$ có nút gốc là r, q_x là đa thức biểu diễn nút x có bậc $d_x = t_x - 1$. $\mathcal{T}_r$ có các nút lá với mỗi nút liên kết với một thuộc tính $att \in A$. Nhóm tác giả xây dựng hàm đệ quy **intermediateKey(S_i)** lấy tham số đầu vào là tập thuộc tính S_i của người dùng i và trả về tập $\mathbf{K}_i = \{k_{i,j} \mid 1 \le j \le m\}$ là tập khóa trung gian của người dùng i. Việc thực hiện hàm **intermediateKey(S_i)** bắt đầu từ nút gốc r tới các nút con x của nó được minh họa trong Hình 2.12.

Thuật toán 4: **KeyGen($\boldsymbol{\beta_x}, \mathbf{K_x}$)**

REPEAT

1: $\{z_1, z_2, \dots, z_N\} \in \{0,1\}^{\ell}$

2: $$\mathbb{A} = \begin{matrix} a_{1,1} & \dots & a_{1,2N} \\ \vdots & \vdots & \vdots \\ a_{N,1} & \dots & a_{N,2N} \end{matrix} \leftarrow a_{i,j} = \begin{cases} 1 & If\ i = j \\ 0 & If\ 1 \le j \le N\ and\ i \ne j \\ H(s_{i,j} || z_j) & If\ N \le j \le 2N \end{cases}$$

3: $Y \leftarrow \mathbb{A}\,.\,Y = 0$

4: $ACV = \left(\sum_{i=1}^{N} k_{i,j}\,.\,e_i^T\right) + Y$ //e_i là vector cơ sở thứ i thuộc $\mathbb{Z}_q^{2N}$

5: $PI_x = \langle ACV_x, (z_1, z_2, \dots, z_N)\rangle$

6: **RETURN: PI** $= \{PI_j \mid att_j, 1 \le j \le m\}$

Hình 2.13: Mã giả thuật toán SecGen

Hình 2.13 mô tả thuật toán **KeyGen($\boldsymbol{\beta_x}, \mathbf{K_x}$)** lấy đầu vào là tập $\boldsymbol{\beta_x} = \{\langle i, s_{i,j}\rangle \mid 1 \le i \le n; j = att(x)\}$ chứa tất cả các giá trị bí mật liên kết với node lá x và tập $\mathbf{K_x} = \{k_{i,j} \mid Usr_i;\ 1 \le i \le N; j = att(x)\}$ là tập các khóa trung gian của tất cả các user có thuộc tính liên kết với nút lá x. Lần lượt thực hiện thuật toán với tất cả các nút lá, trả về **PI** $= \{PI_j \mid att_j, 1 \le j \le m\}$ là tập tất cả các khóa công khai tại các nút lá.

d. Dẫn xuất khóa

Hàm **KeyDer**(β_i, **PI**) nhận đầu vào là tập các bí mật của Usr_i và **PI** để tính toán khóa k từ dưới lên, thông qua các bước:

Gọi $s_{i,j} \in \beta_i$ với $att(x) = j$ là giá trị bí mật của Usr_i tương ứng với nút lá x. Hàm KeyDer$(s_{i,j}, PI_x)$ trả về khóa trung gian $k_x = v_x\,.\,ACV_x$ nếu $att(x) \in \beta_i$ trong

đó v_x là vector dẫn xuất khóa tương ứng với thuộc tính $att_{att(x)}$ và ACV_x là vector kiểm soát truy cập trong PI_x.

Thuật toán 5: **KeyDer($\boldsymbol{\beta_i}$, PI)**

1: **WHILE** $(x = leaf)$

2: $\left(ACV_x, (z_1, z_2, \dots, z_N)\right) \leftarrow PI_x \leftarrow$ **PI**

3: $v_x = \left(0, \dots, 1 \dots, a_{i,N}, \dots, a_{i,2N}\right) \leftarrow \{(z_1, z_2, \dots, z_N), \boldsymbol{\beta_i}\}$

4: $k_x = v_x \,.\, ACV_x$

5: **WHILE** $(x' \,! = r)$

6: $x' = parent(x)$

7: $\mathbf{Q}_{x'} = \{ind_x | parent(x) = x\}; k_{x'} = \{k_i | i \in \mathbf{Q}_{x'}\}$

8: $$\Delta_{i,\mathbf{Q}_{x'}}(y) = \prod_{i \in \mathbf{Q}_{x'}, i \neq j} \frac{y-1}{j-i}$$

9: $$q_{x'}(y) = \sum_{i \in \mathbf{Q}_{x'}} k_i \, \Delta_{i,\mathbf{Q}_{x'}}(y)$$

10: $k_x = q_x(0)$

11: **RETURN:** $k = q_r(0)$

Hình 2.14: Mã giả thuật toán KeyDer

Hàm tiếp tục thực hiện với các nút $x' = parent(x)$, thông qua phép nội suy Lagrange cùng với giá trị của các nút lá x, tính được $q_{x\prime}$ và khóa tại x' là $q_{x'}(0)$. Giả sử người dùng thỏa mãn ít nhất $t_{x\prime}$ nút con của x' và $\mathbf{Q}_{x\prime}$ là tập các chỉ số của $t_{x\prime}$ nút con đó và tập $\{k_i | \in \mathbf{Q}_{x\prime}\}$ là các khóa không $\perp$ tại các nút con của x'

$$\Delta_{i,\mathbf{Q}_{x\prime}}(y) = \prod_{i \in \mathbf{Q}_{x\prime}, i \neq j} \frac{y-1}{j-i}$$

$$q_{x\prime}(y) = \sum_{i \in \mathbf{Q}_{x\prime}} k_i \, \Delta_{i,\mathbf{Q}_{x\prime}}(y)$$

$$k_x = q_x(0)$$

Nếu Usr_i có các giá trị bí mật thỏa mãn cây truy cập, nó lấy được khóa tại nút gốc $k = q_r(0)$. Ngược lại thì trả về giá trị trống. Mã giả thuật toán minh họa trong Hình 2.14.

e. Cập nhật

Phần này, hệ thống tiến hành cập nhật lại các khóa nhóm k và các thông tin công khai **PI** bất kỳ khi nào xảy ra một trong các thay đổi sau trong hệ thống:

- Thay đổi chính sách truy cập **P**
- Người dùng thay đổi các thuộc tính.

Trong các trường hợp này, hệ thống thực hiện hàm KeyGen(P) để cập nhật các thông tin công khai của hệ thống. Các cập nhật này không ảnh hưởng tới những người dùng đã được cấp phát các giá trị bí mật của họ.

2.2.2. Mã hóa dựa trên thuộc tính sử dụng chính sách bản mã

Trong phương pháp mã hóa KP-ABE được sử dụng ở trên, bản mã được liên kết với tập các thuộc tính và khóa của người dùng được liên kết với chính sách truy cập. Tuy nhiên, nhược điểm của nó là người mã hóa không kiểm soát được ai có quyền truy cập vào dữ liệu mã hóa, mà chức năng này trao cho một bên thứ ba và bên mã hóa phải tin tưởng rằng họ sẽ cấp khóa cho người dùng phù hợp.

Trong phương pháp mới này, CP-ABE (Ciphertext-Policy Attribute-Based Encryption) [22], thông qua cặp PK – MK, người mã hóa hoàn toàn quyết định ai là người có thể đọc được bản mã. Khi mã hóa, họ chỉ định một cấu trúc truy cập lên nó qua các thuộc tính liên quan, và do đó, cấu trúc truy cập được "nhúng" vào trong bản mã. Song song với đó, chi phí về mã hóa và giải mã cũng tăng lên đáng kể.

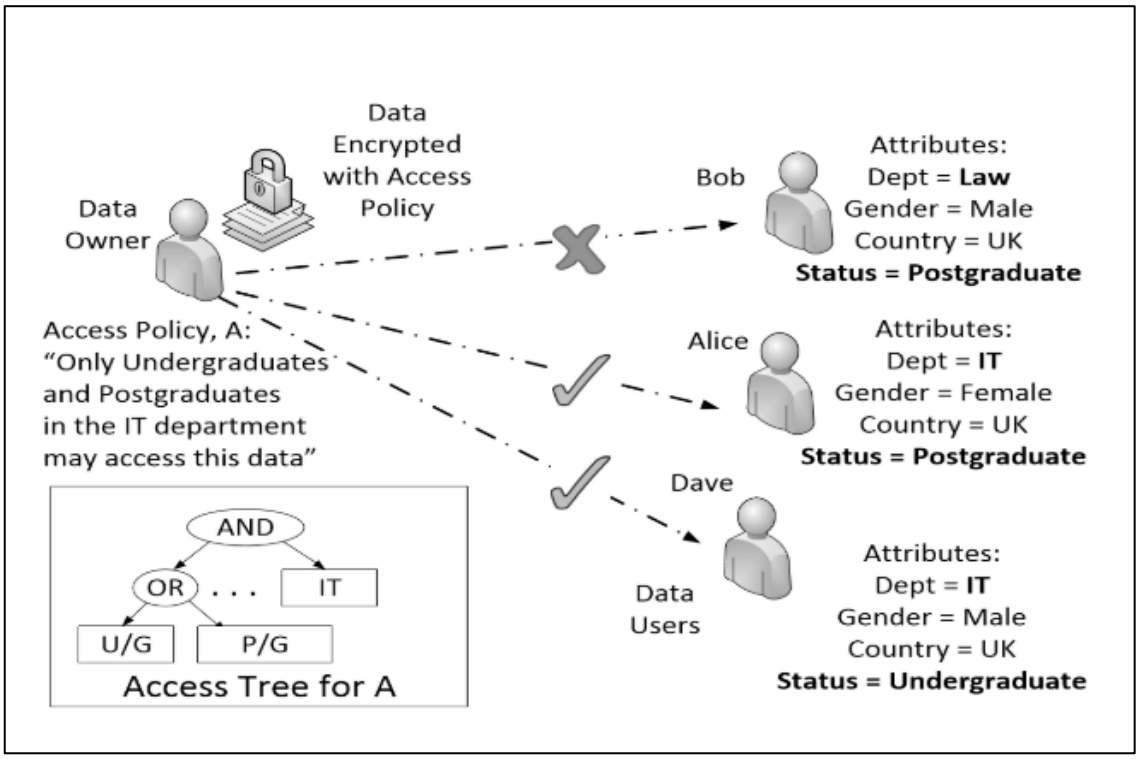
Gọi $\mathbb{G}_0$ và $\mathbb{G}_1$ là hai nhóm nhân hữu hạn có cấp là số nguyên tố p. g là phần tử sinh từ $\mathbb{G}_0$. e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ là ánh xạ song tuyến tính nếu:

- $e(x_1 . x_2, z) = e(x_1, z). e(x_2, z)$
- $e(x, y_1 . y_2) = e(x, y_1) . e(x, y_2)$
- Không suy biên: $e(g, g) \neq 1$

Từ định nghĩa như vậy, ta có tính chất của ánh xạ song tuyến tính e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$:

$\forall\, u,\, v \in \mathbb{G}_0$ và $a,\, b \in \mathbb{Z}_p$. Ta có $e\,(u^a, v^b) = e(u,\, v)^{ab}$

Hình 2.15 mô tả tổng quan về CP-ABE với ngữ cảnh: chủ sở hữu dữ liệu DO muốn chia sẻ một số dữ liệu riêng tư cho người dùng DU được lưu trữ trên cơ sở dữ liệu đám mây. Thay vì cung cấp quyền truy cập riêng lẻ cho từng người dùng, DO cho phép DU truy cập vào dữ liệu khi và chỉ khi họ có thông tin về tập thuộc tính phù hợp với chính sách của DO. Nếu DU đáp ứng các chính sách truy cập, họ có thể xem được dữ liệu mà DO đã chia sẻ.

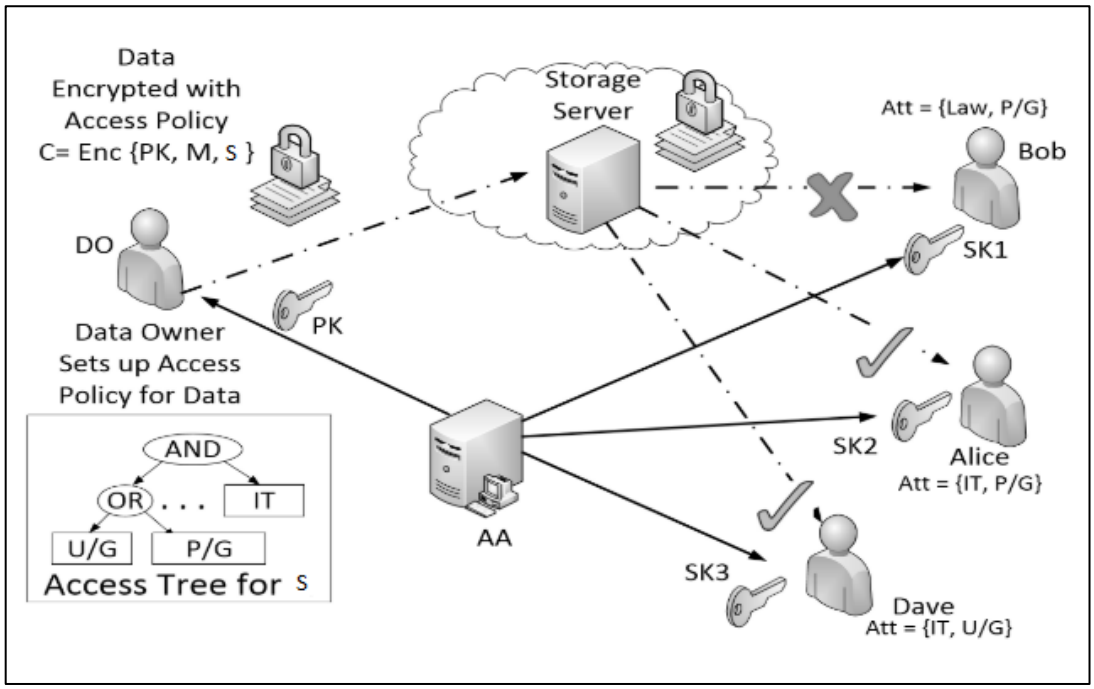


Hình 2.15: Tổng quan về CP-ABE [23]

Phương pháp CP-ABE nhóm tác giả đề cập đến trong khóa luận này bao gồm 5 thuật toán cơ bản được minh họa trong Hình 2.16. Khi DO bắt đầu thực hiện thiết lập truy cập với chính sách truy cập $\boldsymbol{S}$ bao gồm tất cả các thuộc tính của người dùng, cho dữ liệu **M** của mình. DO mã hóa cả dữ liệu M và tập truy cập liên quan đến nó $\boldsymbol{S} \subseteq \boldsymbol{A}$ thông

qua khóa công khai PK. DU được chỉ định một khóa riêng của họ, được liên kết với tập thuộc tính $\boldsymbol{S_i} \subset \boldsymbol{A}$. PK và SK đều được tạo từ cùng một MasterKey bởi AA. DU có thể giải mã C bằng SK khi và chỉ khi danh sách thuộc tính $\boldsymbol{S_i}$ được liên kết với SK đáp ứng tiêu chí của quyền truy cập được gắn cho M. Có 5 thuật toán được sử dụng trong quá trình này bao gồm:

- Setup$(\kappa) \rightarrow PK, MK$
- KeyGen$(MK, \mathbf{S}_i) \rightarrow SK_i$
- Encrypt$(PK, M, \boldsymbol{S}) \rightarrow$ CT
- Decrypt$(\text{CT}, SK_i) \rightarrow M$
- Delegate$\left(SK, \tilde{S}_i\right) \rightarrow \widetilde{SK}_i$



Hình 2.16: Hiện thực CP-ABE [23]

a. Thiết lập

Thuật toán 6: **Setup**($\boldsymbol{\kappa}$)

1: $g \leftarrow \mathbb{G}_0;\ \alpha, \beta \in \mathbb{Z}_q$

2: $h = g^{\beta}, f = g^{1/\beta}$

3: **RETURN:**

4 $PK = \{\mathbb{G}_0, g, h, e(g,g)^{\alpha}, f\ (*)\}$ //f chỉ sử dụng khi ủy quyền

5 $MK = (\beta, g^{\alpha})$

Hình 2.17: Mã giả thuật toán Setup

Như đã định nghĩa ở trên, gọi $\mathbb{G}_0$ là nhóm song tuyến của số nguyên tố q, g là phần tử sinh từ tập $\mathbb{G}_0$. e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ là hàm song tuyến tính. Tham số bảo mật κ xác định kích thước $\mathbb{G}_0$. Định nghĩa hàm Lagrange $\Delta_{i,\mathbf{Q}}(x) = \prod_{j \in \mathbf{Q}, j \neq i} \frac{x-j}{i-j}$ với $i \in \mathbb{Z}_q$ và $\mathbf{Q}$ là tập các phần tử trong $\mathbb{Z}_q$.

b. Tạo khóa

Thuật toán 7: **KeyGen**($\boldsymbol{MK}, \mathbf{S_i}$)

1: $r_i, \leftarrow \mathbb{Z}_q$

2: $D_i = \sqrt[\beta]{g^{\alpha}.g^{r_i}} = g^{(\alpha + r_i)/\beta}$

3: **FOR EACH** $(att_j \in \mathbf{S}_i)$

4: $r_{i,j} \leftarrow \mathbb{Z}_q$

5: $D_{i,j} = g^{r_i}.H(att_j)^{r_{i,j}}$

6: $D'_{i,j} = g^{r_{i,j}}$

7: **RETURN**

8: $SK_i = \{D_i, D_{i,j}, D'_{i,j} \mid 1 \leq j \leq j_i\}$

Hình 2.18: Mã giả thuật toán KeyGen

Hàm tạo khóa KeyGen($MK, \mathbf{S}_i$) lấy đầu vào là tập thuộc tính $\mathbf{S}_i$ của người dùng Usr_i và khóa MK, trả về một khóa SK_i là khóa chứng thực người dùng với tập thuộc tính liên kết với nó là $\mathbf{S}_i$.

c. Ủy quyền

Hàm ủy quyền Delegate($SK_i, \tilde{\boldsymbol{S}}$) được sử dụng bởi một người dùng hợp lệ. Người này chia sẻ một số quyền của mình cho người dùng khác thông qua tập thuộc tính $\tilde{\boldsymbol{S}} \subseteq \mathbf{S}_i$. Thuật toán được mô tả trong Hình 2.19.

Thuật toán 8: **Delegate**($\boldsymbol{SK_i}, \tilde{\boldsymbol{S}}$)

1: $\tilde{r} \leftarrow \mathbb{Z}_q$

2: $\widetilde{D} = D_i \,.\, f^{\tilde{r}}$

3: **FOR EACH** ($att_k \in \tilde{\boldsymbol{S}}$)

4: $\tilde{r}_k \leftarrow \mathbb{Z}_q$

5: $\widetilde{D}_k = D_{i,k}.\, g^{\tilde{r}}.\, H(att_k)^{\tilde{r}_k}$

6: $\widetilde{D'}_k = D'_{i,k} \,.\, g^{\tilde{r}_k}$

7: **RETURN**

8: $\widetilde{SK} = \{\widetilde{D}, \widetilde{D}_k, \widetilde{D'}_k \mid \forall\, att_k \in \tilde{\boldsymbol{S}}\}$

Hình 2.19: Mã giả thuật toán ủy quyền

d. Mã hóa

Thuật toán mã hóa Encrypt($PK, M, \mathcal{T}$) mã hóa dữ liệu M dưới chính sách truy cập quy định tại $\mathcal{T}$. Các thành phần của $\mathcal{T}$ giống như đã trình bày tại phần 2.2. Thuật toán được trình bày trong Hình 2.20.

e. Giải mã

Hàm giải mã Decrypt(CT, SK_i) nhận đầu vào là bản mã hóa CT và khóa bí mật SK_i của người dùng Usr_i, trả về dữ liệu M nếu tập SK_i liên kết với tập thuộc tính đã sử dụng để mã hóa CT.

Định nghĩa hàm đệ quy DecryptNode(CT, SK_i, x) lấy đầu vào là bản mã CT, khóa bí mật SK_i liên kết với tập thuộc tính $\mathbf{S}_i$và x là một nút trong $\mathcal{T}$. Hàm DecryptNode(CT, SK_i, x) định nghĩa trong Hình 2.21

Thuật toán 9: **Encrypt**$(\boldsymbol{PK}, \boldsymbol{M}, \boldsymbol{\mathcal{T}})$

1: $k \leftarrow \mathbb{Z}_q$

2: $\tilde{C} = M \,.\, e(g,g)^{\alpha k}, C = h^k$

3: $q_r(0) = k; d_r = t_r - 1$

4: $\mathbf{R_r} = \{(x_t; y_t) \,|(\, x_t, y_t \in \mathbb{Z}_q; 1 \leq t \leq d_r\}$

5: $q_r = \sum_{i=1}^{t_r} q_r(x_i) \,.\, \prod_{\substack{j=1 \\ j \neq i}}^{t_r} \frac{x - x_j}{x_i - x_j} \,|\, \{q_r(x_i) = y_i; (x_i, y_i) \in \mathbf{R_r}\}$

6: **IF** $(x \,!= leaf)$

7: $q_x(0) = q_{parent(x)}(index(x)); d_x = t_x - 1$

8: $\mathbf{R_x} = \{(x_t; y_t) \,|(\, x_t, y_t \in \mathbb{Z}_q; 1 \leq t \leq d_x\}$

9: $q_x = \sum_{i=1}^{t_x} q_x(x_i) \,.\, \prod_{\substack{j=1 \\ j \neq i}}^{t_x} \frac{x - x_j}{x_i - x_j} \,|\, \{q_x(x_i) = y_i; (x_i, y_i) \in \mathbf{R_x}\}$

10: **ELSE**

11: $C_x = g^{q_x(0)}; C'_x = H\big(att(x)\big)^{q_x(0)}$

12: **RETURN:**

13: $CT = \big\{\mathcal{T}, \tilde{C}, C, C_x, C'_x \,\big|x \; is \; leaf \; node \; of \; \mathcal{T}, att(x) \in \boldsymbol{A}\}$

Hình 2.20: Thuật toán mã hóa

Hàm 1: **DecryptNode**$(\boldsymbol{CT}, \boldsymbol{SK_i}, \boldsymbol{x})$

IF $(x = leaf)$

  $j = att(x)$

  IF $(j \in \mathbf{S}_i)$

    DecryptNode$(CT, SK_i, x) = \frac{e(D_{i,j}, C_x)}{e(D'_{i,j}, C'_x)} = e(g,g)^{r_i \cdot q_x(0)}$

  ELSE

    DecryptNode$(CT, SK_i, x) = \perp$

ELSE

  FOR EACH $(z \in \mathcal{T} \mid parent(z) = x)$

  $F_z =$ DecryptNode(CT, SK_i, z)

  IF $F_z \neq \perp$

    $\mathcal{S}_x \mathrel{+}= F_z$

  IF $(size - of(\mathcal{S}_x) = t_x)$

$$F_x = \prod_{z \in \mathcal{S}_x} F_z^{\Delta_{ind, \mathcal{S}'_x}(0)} \quad where \begin{cases} ind = index(z) \\ \mathcal{S}'_x = \{index(z); z \in \mathcal{S}_x \end{cases}$$

$$= \prod_{z \in \mathcal{S}_x} \left(e(g,g)^{r_i \cdot q_z(0)}\right)^{\Delta_{ind, \mathcal{S}'_x}(0)}$$

$$= \prod_{z \in \mathcal{S}_x} \left(e(g,g)^{r_i \cdot q_{parent(z)}(index(z))}\right)^{\Delta_{ind, \mathcal{S}'_x}(0)}$$

$$= \prod_{z \in \mathcal{S}_x} \left(e(g,g)^{r_i \cdot q_x(ind)}\right)^{\Delta_{ind, \mathcal{S}'_x}(0)} = e(g,g)^{r_i q_x(0)}$$

  ELSE

    $F_x = \perp$

  RETURN F_x

Hình 2.21: Hàm giải mã nút

Nếu tập $\mathbf{S}_i$ thỏa mãn chính sách truy cập trong $\mathcal{T}$, gán $A = \text{DecryptNode}(CT, SK_i, r) = e(g,g)^{r_i q_r(0)} = e(g,g)^{r_i k}$. Ta có dữ liệu được giải mã là

$$M = \tilde{C} / \left(\frac{e(C, D_i)}{A} \right) = \tilde{C} / \left(\frac{e\left(h^k, g^{\frac{(\alpha + r_i)}{\beta}} \right)}{e(g,g)^{r_i k}} \right)$$

CHƯƠNG 3. **MÃ HÓA DỮ LIỆU VÀ CÁC THAO TÁC TRÊN CƠ SỞ DỮ LIỆU MÃ HÓA**

3.1. Tổng quan về mã hóa dữ liệu và các phương pháp tính toán trên dữ liệu được mã hóa trong thực tế

Với những ưu điểm của cơ sở dữ liệu đám mây như: đảm bảo dữ liệu luôn sẵn sàng, tối ưu chi phí về cơ sở hạ tầng, chi phí bảo trì, linh hoạt về khả năng phát triển, mở rộng hoặc thu hẹp, đảm bảo tài nguyên được sử dụng tối ưu. Bên cạnh đó, người dùng cũng thích sử dụng các dịch vụ lưu trữ đám mây để giảm bớt các chi phí về bảo trì cũng như chi phí lưu trữ dữ liệu cục bộ. Cũng như người dùng có thể truy cập dữ liệu của mình từ bất kỳ đâu và bất kỳ lúc nào thay vì phải sử dụng máy chuyên dụng.

Mặc dù lưu trữ đám mây có nhiều những ưu điểm vượt trội như trên, nhưng vẫn còn rất nhiều vấn đề về cần lo ngại về bảo mật khác nhau. Môi trường đám mây được xem là không tin cậy cho những dữ liệu riêng tư có tính nhạy cảm: dữ liệu không chỉ bị truy cập trái phép mà còn có thể bị lạm dụng từ nhà quản lý cung cấp dịch vụ hoặc bị đánh cắp bởi những kẻ tấn công hệ thống. Do vậy ngoài những cơ chế bảo mật được cung cấp sẵn từ nhà cung cấp dịch vụ chủ sở hữu dữ liệu cần có những cơ chế bảo mật riêng để đảm bảo thông tin lưu trữ, tương tác và trao đổi được an toàn. Một trong những giải pháp được sử dụng là mã hóa dữ liệu trước khi lưu trữ tại đám mây và thay đổi phương pháp tương tác truyền thống trên dữ liệu rõ bằng các phương pháp tương tác trên dữ liệu mã hóa.

Trong thực tế có rất nhiều công cụ để thực hiện tính toán trên dữ liệu mã, một vài công cụ sẽ cung cấp hiệu năng đầy hứa hẹn trong một số trường hợp cụ thể. Những công cụ hoặc phương pháp này dựa trên ba yếu tố: chức năng, bảo mật và hiệu suất. Tuy nhiên không một công cụ nào có thể đáp ứng đủ ba yếu tố trên để động đơn lẻ đẻ nhằm đáp ứng được trên hệ thống dữ liệu được mã hóa ngày nay. Về phân loại, các công cụ và phương pháp tính toán trên dữ liệu mã hóa được chia làm hai loại như sau:

- Những công cụ gần như không làm rò rỉ gì về dữ liệu (Tool with no leakage). Bao gồm các lược đồ mã hóa đồng cấu - Homomorphic Encryption (HE)

- Những công cụ chỉ để lộ một chức năng xác định cụ thể nào đó của dữ liệu hay còn gọi là những công cụ kiểm soát rò rỉ dữ liệu (Tool with controlled leakage). Bao gồm:

+ Functional encryption

+ Garbled circuits

+ Secure multi-party computation (MPC)

+ Những công cụ đặc biệt khác:

- Searchable encryption
- Order-preserving encryption (OPE)
- Deterministic encryption

3.1.1. Mã hóa đồng cấu - Homomorphic Encryption (HE)

a. Tổng quan về mã hóa đồng cấu

Mã hóa đồng cấu - Homomorphic Encryption (HE) cho phép tính toán trực tiếp trên dữ liệu mã hóa mà không cần thao tác giải mã trước đó. Điều kiện quan trọng nhất trong mã hóa đồng hình là giá trị đạt được sau khi giải mã kết quả thu được bằng cách áp dụng tính toán trên bản mã phải giống với giá trị đạt được bằng cách áp dụng các phép tính tương tự trên bản rõ [24].

Mã hóa đồng cấu được ứng dụng trong nhiều lĩnh vực, chẳng hạn như tài chính/kinh doanh, chăm sóc sức khỏe và bất kỳ lĩnh vực nào hoạt động với dữ liệu nhạy cảm.

b. Định nghĩa

Một function $g: A \rightarrow B$ được gọi là đồng cấu qua phép toán $*$ nếu thỏa mãn điều kiện sau đây:

$$g(x_1) * g(x_2) = g(x_1 * x_2), \forall x_1, x_2 \in A$$

Bên cạnh một hệ thống mã hóa chung bao gồm ba thuật toán: tạo khóa, mã hóa, giải mã. Mã hóa đồng cấu còn có thêm một thuật toán được gọi là evaluation, được ký hiệu là Eval. Đầu vào và đầu ra của thuật toán Eval dưới dạng mã hóa. Trong thuật toán Eval một hàm g thực hiện các phép tính trên bản mã $c_1$và c_2 mà không cần truy cập vào bản rõ m_1 và m_2 có tính chất sau:

$$Dec\left(key_{priv}, Eval_g(key_{eval}, c_1, c_2)\right) = f(m_1, m_2)$$

c. Phân loại

Mã hóa đồng cấu được phân làm 3 nhóm như sau:

- Partial homomorphic encryption (PHE): Các lược đồ ở nhóm này chỉ hỗ trợ cho một phép toán (cộng hoặc nhân) trên dữ liệu mã hóa mà không giới hạn số lần thực hiện. VD: RSA, Goldwasser-Micali, El-Gamal. PHE là nền tảng cho các lược đồ homomorphic khác.

- Somewhat homomorphic encryption (SWHE): Các lược đồ ở nhóm này hỗ trợ cả hai phép toán (cộng và nhân) trên dữ liệu mã hóa nhưng giới hạn số lần thực hiện. SWHE là tiền thân của PHE.

- Fully homomorphic encryption (FHE): Các lược đồ ở nhóm này hỗ trợ cả hai phép toán (cộng và nhân) trên dữ liệu mã hóa mà không giới hạn số lần thực hiện.

3.1.2. Mã hóa có thể tìm kiếm - Searchable encryption (SE)

a. Tổng quan về mã hóa có thể tìm kiếm

SE là một kỹ thuật mã hóa cho phép lưu dữ liệu được mã hóa trên máy chủ đám mây từ nhà cung cấp dịch vụ bên thứ ba không đáng tin cậy, trong khi đồng thời cho phép người dùng áp dụng các hoạt động tìm kiếm trực tiếp trên dữ liệu mã hóa một cách an toàn và bảo mật. SE có thể được coi là một lược đồ Fully homomorphic encryption [24].

Ví dụ về searchable encryption:

- Data owner A: Lưu tập tài liệu lưu trữ trên Server.

- Data user B: Được cấp quyền truy cập tập tài liệu.

- Để đảm bảo bảo mật data owner A mã hóa tài liệu bằng public key data user B và lưu trữ trên Server.

- Data user B: Trong trường hợp này chỉ có quyền tìm kiếm trong tài liệu (định dạng mã hóa) hoặc đọc tài liệu (sau khi truy xuất từ máy chủ và được giải mã).

- Nếu data user B: muốn trích xuất từ máy chủ bất kỳ tài liệu nào có chứa một từ khóa cụ thể như "programming".

- Data user B: xây dựng giá trị trapdoor (dựa trên từ truy vấn "programming" cùng với private key của data user B) và gửi giá trị trapdoor đến server.

- Server: Chạy thuật toán tìm kiếm được cung cấp bởi searchable encryption scheme sau đó gửi kết quả (dưới dạng mã hóa đến data user B).

b. Mô hình của searchable encryption

Searchable encryption bao gồm các thực thể (entities) và các thuật toán (algorithms).

Trong hệ thống sử dụng searchable encryption scheme, những thực thể/ đối tượng sẽ sau sẽ tham gia vào toàn bộ quá trình:

- Data owner: Chủ sử hữu dữ liệu, người được coi là 1 bên đáng tin cậy, với số lượng tài liệu $n = D_1, \dots, D_n$, được đặc trưng bởi các từ khóa (keywords không phải metadata/ siêu dữ liệu). Tài liệu và từ khóa sẽ được lưu trữ ở các cơ sở dữ liệu cho thuê. Trước khi lưu trữ, các tài liệu và từ khóa thường được sắp xếp theo cấu trúc gọi là cấu trúc chỉ mục – index structure trên máy chủ và mã hóa bởi chủ sở hữu dữ liệu bằng cách sử dụng thuật toán mã hóa của lược đồ searchable encryption.

- Data user: Người dùng dữ liệu, được cấp phép quyền để thực hiện quá trình tìm kiếm. Sử dụng từ khóa cần tìm kiếm người dùng dữ liệu sẽ tạo ra một giá trị gọi là "Trapdoor", giá trị này sẽ được sử dụng khi tìm kiếm trên dữ liệu mã hóa. Lưu ý rằng: chủ sở hữu dữ liệu có thể là một người dùng dữ liệu.

- **Server**: Máy chủ lưu trữ dữ liệu mã hóa và thực hiện thuật toán tìm kiếm dựa trên giá trị "Trapdoor" nhận được từ data user. Máy chủ được xem là semi-trusted hay honest-but-curious, có nghĩa là máy chủ sẽ thực hiện thuật toán tìm kiếm theo chỉ dẫn nhưng vẫn có thể phân tích dữ liệu được cung cấp.

c. Yêu cầu về bảo mật của searchable encryption

Trong lược đồ searchable encryption, bên cạnh việc đảm bảo tính bảo mật của các tài liệu và từ khóa lưu trữ trên máy chủ còn có các từ khóa truy vấn. Ngoài ra một vài mục sau đây cũng cần được bảo vệ:

- **Search pattern**: là thông tin được tìm thấy từ hai từ hai kết quả truy vấn khác nhau thuộc về cùng một từ khóa.

- **Access pattern**: là tập hợp kết quả các tài liệu từ trapdoor tương ứng với từ khóa đã cho.

Gần đây một số yêu cầu bảo mật là forward và backward privacy của lược đồ dynamic searchable cho phép cập nhật, thêm, xóa được áp dụng trực tiếp trên các tài liệu hoặc từ khóa trực tiếp trên máy chủ mà không cần giải mã đã được đưa ra. Cụ thể:

- **Backward privacy**: ngăn cách hoạt động tìm kiếm làm rò rỉ các phần tử phù hợp sau khi chúng đã bị xóa.

- **Forward privacy**: các hoạt động cập nhật (addition và deletion) không được liên kết với các truy vấn tìm kiếm trước đó

d. Phân loại và thuật toán của các lược đồ searchable encryption

Lược đồ searchable encryption được phân thành hai nhóm như sau:

- Symmetric searchable encryption (SSE): chỉ sử dụng một khóa cho quá trình mã hóa và giải mã.

- Public encryption with keywords search (PEKS): sử dụng hai khóa cho quá trình mã hóa và giải mã. Public key để mã hóa dữ liệu, private (hoặc secret) key để giải mã nội dung mã hóa.

Thuật toán lược đồ Symmetric searchable encryption (SSE)

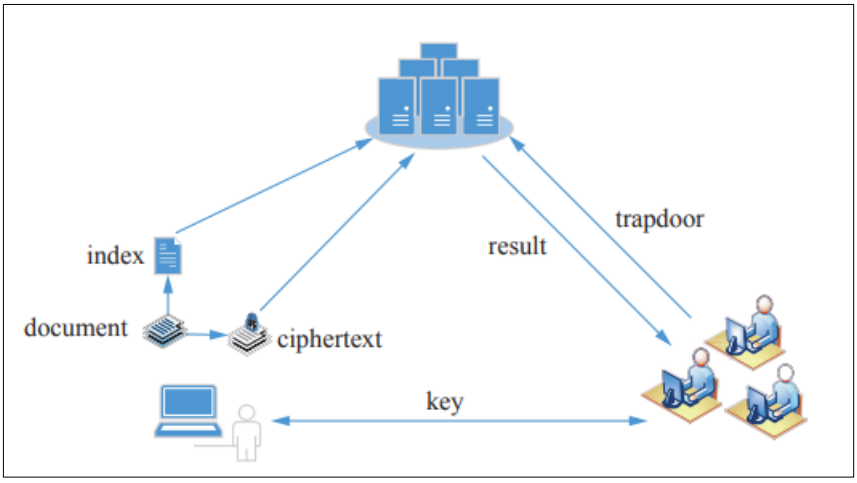
Một lược đồ Symmetric searchable encryption bao gồm bốn thuật toán:

- $Keygen\ (1^k) \rightarrow K$: Data owner chạy thuật toán này. Đầu vào là một tham số bảo mật k, đầu ra là một khóa bí mật K.

- $BuildIndex(K, D) \rightarrow I$: Data owner chạy thuật toán này. Đầu vào là khóa bí mật K và tập tài liệu D, đầu ra là cấu trúc chỉ mục I.

- $Trapdoor(K, w) \rightarrow\ T_w$: Data user chạy thuật toán này. Đầu vào là khóa bí mật K và từ khóa truy vấn w, đầu ra là giá trị trapdoor T_w.

- $Search(I, T_w) \rightarrow\ D_w$: Server chạy thuật toán này. Đầu vào là chỉ mục I và giá trị trapdoor T_w, đầu ra là tập tài liệu D_w chứa từ khóa w tương ứng.



Hình 3.1: Mô hình của lược đồ SSE [25]

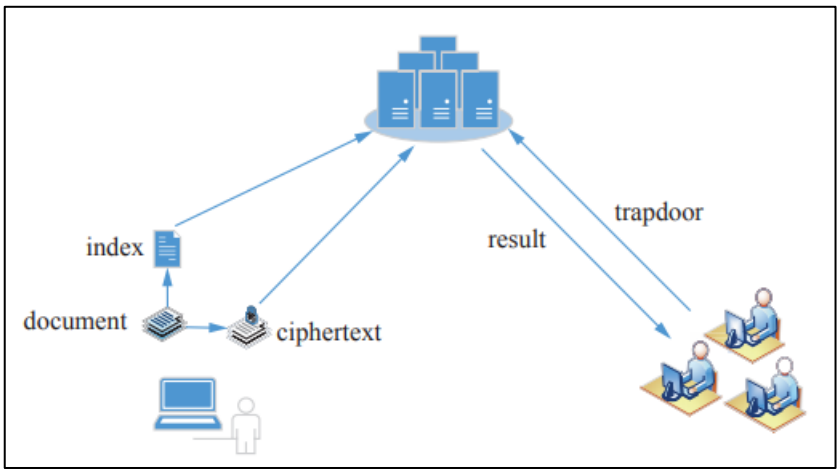
Thuật toán lược đồ Public encryption with keywords search (PEKS):

- $Keygen\ (1^k) \rightarrow K_{pub}, K_{priv}$

- $PEKS(K_{pub}, W) \rightarrow C_w$

- $Trapdoor(K_{priv}, W) \rightarrow\ T_w$

- $Test(K_{pub}, C_w, T_w) \rightarrow yes\ or\ no$



Hình 3.2: Mô hình lược đồ PEKS [25]

e. Các lược đồ Searchable symmetric encryption (SSE)

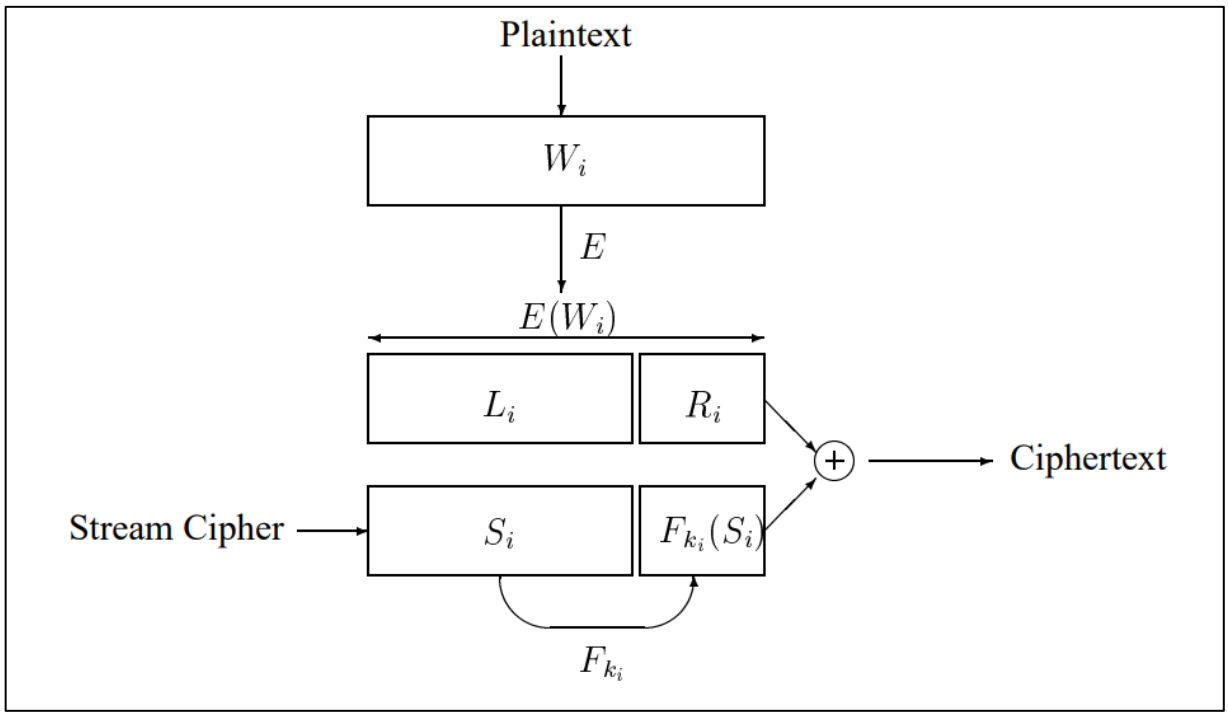
Các lược đồ SSE có thể được phân loại như sau:

- Single keyword search
    + SSE schemes with sequential scan (Quét tuần tự)
    + SSE schemes with secure index (Chỉ mục an toàn)
    + Dynamic SSE scheme
- Fuzzy keyword search
- Conjunctive keyword search
- Ranked and verifiable keyword search

3.2. Một vài lược đồ mã hóa đối xứng có thể tìm kiếm - Searchable symmetric encryption (SSE)

3.2.1. Tìm kiếm từ khóa đơn

Tìm kiếm từ khóa đơn (Single keyword search) là lược đồ SSE với quét tuần tự, Song và cộng sự [26] đã đề xuất lược đồ SSE đầu tiên.



Hình 3.3: Thuật toán quét tuần tự [26]

Ý tưởng của lược đồ này chính là bản mã thu được bằng cách XOR từng từ khóa trong bản rõ với một chuỗi các bit giả ngẫu nhiên. Do đó, cho phép tìm kiếm trực tiếp bên bản mã.

Lược đồ bao gồm 3 bước: encryption, search, decryption:

- **Encryption**: mã hóa được thực hiện bởi người dùng. Giả sử người dùng muốn mã hóa một tài liệu chứa một chuỗi các từ khóa $W_1, \ldots, W_l$. Để mã hóa từ khóa từng từ khóa W_i, bao gồm các bước sau:

+ $X_i = E_{k''}(W_i)$: người dùng mã hóa W_i bằng cách sử dụng hàm E cùng với $key\ k''$ và đạt được bản mã X_i có độ dài n bits.

+ Sau đó X_i được chia thành hai phần: phần bên trái L_i và phần bên phải R_i. Trong đó L_i là $n - m$ bits đầu liên và R_i là m bits còn lại.

+ Cuối cùng người dùng tạo ra một chuỗi giá trị $S_1, \ldots, S_i$ trong đó S_i

có độ dài $n - m$ bits. Để mã hóa toàn bộ n bits X_i. Người dùng lấy giá trị S_i và đặt $T_i = \langle\left(S_i, F_{k_i}(S_i)\right)\rangle$. Đầu ra là bản mã $C_i = X_i \oplus T_i$ trong đó $k_i = f_{k'}(L_i)$. Người dùng xuất tất cả các bản mã C_i và gửi đến máy chủ.

- **Search**: tìm kiếm được thực hiện bởi máy chủ. Giả sử người dùng muốn tìm kiếm tài liệu chứa từ khóa W_i, người dùng sẽ tính toán $X_i = E_{k''}(W_i)$ và $k_i = f_{k'}(L_i)$. và gửi $\langle X_i, k_i\rangle$ đến máy chủ. Máy chủ thực hiện tìm kiếm X_i trong bản mã bằng cách kiểm tra $C_i \oplus X_i$ dưới dạng $\langle\left(S_i, F_{k_i}(S_i)\right)\rangle$ với một vài S_i. Nếu C_i tồn tại, máy chủ sẽ gửi toàn bộ tài liệu dưới dạng bản mã có chứa từ khóa truy vấn đến người dùng

- **Decryption**: giải mã được thực hiện bởi người dùng. Với mỗi bản mã C_i, người dùng tạo ra S_i bằng cách sử dụng pseudorandom generator. Sau đó $S_i \oplus (n - m)$ bits đầu tiên của C_i và đạt được L_i với L_i người dùng có thể tính toán k_i và khôi phục lại W_i.

Trong lược đồ này, bản rõ và từ khóa truy vấn được bảo mật. Lợi ích của một lược đồ SSE với quét tuần tự là dễ dàng cập nhật. Tuy nhiên nhược điểm quét tuần tự mang lại là không hiệu quả với cơ sở dữ liệu lớn do thời gian tìm kiếm tuyến tính theo độ dài của bộ tài liệu và máy chủ phải quét toàn bộ bản mã của tài liệu.

3.2.2. Tìm kiếm từ khóa đơn với quyền riêng tư chuyển tiếp

Searchable symmetric encryption đã được áp dụng rộng rãi cho các truy vấn trong cơ sở dữ liệu được mã hóa trong thực tế. Mặc dù SSE mạnh mẽ và giàu tính năng nhưng vẫn luôn bị hạn chế bởi sự cố rò rỉ thông tin. Một vài cuộc tấn công đã chỉ ra rằng forward privacy ngăn chặn rò rỉ từ các hoạt động cập nhật, giờ đây đã trở thành một yêu cầu cơ bản đối với bất kỳ lược đồ SSE mới nào được thế kế. Tuy nhiên, các hoạt động tìm kiếm sau đó vẫn có thể rò rỉ một lượng thông tin đáng kể. Để tăng cường bảo mật hơn nữa, một định nghĩa mở rộng về forward privacy đã được đề là "forward search privacy" (FsP). Trực quan, FsP yêu cầu các hoạt động tìm kiếm trên các tài liệu mới được thêm vào không làm rò rỉ bất kỳ thông tin nào về các truy vấn trước đây [27].

Khái niệm bảo mật nâng cao này đặt ra thách thức mới đối với việc thiết kế SSE. Để giải quyết những thách thức này kỹ thuật Hidden pointer (HPT) đã được phát triển và một lược đồ SSE mới được đề xuất gọi là Khons. Khons đáp ứng FsP và khái niệm forward privacy ban đầu.

a. Sự cần thiết của forward privacy

Mã hóa xác định (deterministic encryption) sử dụng trong SSE giúp các máy chủ độc hại (malicious server) dễ dàng quan sát các truy vấn lặp lại và các thông tin khác. Những rỏ rỉ này được mô hình hóa dưới dạng search pattern (sự lặp đi lặp lại mẫu (pattern) trong các truy vấn tìm kiếm - search queries), size pattern (số lượng kết quả tìm kiếm) và access pattern (cách truy cập vào các dữ liệu hoặc chỉ mục - index được mã hóa).

Nói chung, những rò rỉ này có thể được loại bỏ bằng cách sử dụng oblivious RAM (ORAM). Tuy nhiên ORAM [28] thường mang lại chi phí tính toán và bang thông cao cho mỗi từ khóa tìm kiếm. Do đó, thực tế một SSE phải cho phép rò rỉ thông tin để đổi lấy hiệu quả ở mức chấp nhận được. Thật không may, những rò rỉ này đã bị lạm dụng để tấn công các lược đồ SSE theo nhiều cách khác nhau.

Trong năm 2016 Zhang và các cộng sự [29] đề xuất cuộc tấn công file-injection . Cuộc tấn công này giả định rằng kẻ xấu có thể tiêm vào các tệp (inject files) tức là tạo ra một bộ tài liệu và lừa người dùng mã hóa chúng. Bằng cách tiêm vào các tệp đã được lựa chọn kỹ cẩn thận, kẻ xấu có thể khôi phục các từ khóa cần được giữ bí mật, từ các mã tìm kiếm được gửi lên bởi người dùng. Cuộc tấn công rất hiệu quả và chỉ yêu cầu một số lượng nhỏ tài liệu được thêm vào.Vấn đề được nhấn mạnh ở cuộc tấn công này là khái niệm bảo mật được sử dụng trong quá khứ là quá yếu. Cụ thể hơn nó cho phép kẻ xấu đạt được thông tin về các từ khóa được truy vấn trong quá khứ bằng cách liên kết mã thông báo đã được gửi trước đây với các tài liệu vừa được cập nhật. Cuộc tấn công yêu cầu biện pháp nghiêm ngặt hơn đối với việc rò thông tin trong SSE và biến forward privacy trở thành "đường cơ sở" (baseline) cho các lược đồ SSE mới được phát triển.

b. Giới hạn bảo mật của forward privacy

Hiện nay, forward privacy và backward privacy đã trở thành một yêu cầu bảo mật cơ bản cho SSE mà không cần ORAM. Tuy nhiên, forward privacy vẫn chưa đủ thỏa mãn và vẫn còn khả năng để cải thiện bảo mật.

Vấn đề của forward privacy là các bản cập nhật mới chỉ có thể hủy liên kết với các truy vấn trước đó cho đến khi một truy vấn tìm kiếm được thực hiện. Truy vấn tìm kiếm liên kết tất cả các các cập nhật phù hợp với cùng một từ khóa. Đây là lí do tại sao forward privacy có thể chống lại cuộc tấn công file-injection nhưng không thể chống lại statistical inference. Những cuộc tấn công statistical inference này dựa trên một tập hợp lớn các thông tin về hành vi truy vấn giống nhau. Nếu các tài liệu mới được thêm vào vẫn không thể liên kết được sau các truy vấn tìm kiếm, kẻ xấu sẽ khó suy ra các từ khóa đang được truy vấn, và có thể phòng thủ trước các cuộc tấn công statistical.

Ở một mức độ nào nó, forward privacy hiện tại chỉ liên quan đến rò rỉ thông tin do gây ra bởi các hoạt động cập nhật và có thể được coi là "forward update privacy" FuP. Khái niệm bảo mật nghiêm ngặt hơn nên xét thông tin bị rò rỉ qua các hoạt động tìm kiếm. Nếu một lược đồ SSE đạt được forward update privacy và hoạt động tìm kiếm của lược đồ SSE này trên các tài liệu mới được cập nhật hoặc trên các tài liệu trong một khoảng thời gian mà không làm rò rỉ thông tin truy vấn trong quá khứ thì nó sẽ đạt được một khái niệm bảo mật mới mà các tác giả gọi là "forward search privacy" FsP.

c. Những thách thức về thiết kế lược đồ forward search privacy

Về cơ bản, forward search privacy thành 2 loại tương ứng:

- Weak forward search privacy
- Strong forward search privacy

Đối với weak forward search privacy, thách thức lớn nhất để thiết kế một lược đồ SSE hoàn toàn mới là để có thể cân bằng giữa bảo mật và hiệu quả. Xét đến tính hiệu quả, hầu hết các lược đồ SSE chỉ sử dụng inverted index để ánh xạ một từ khóa đến một tập hợp các tài liệu có chứa từ khóa này.

Về mặt khái niệm, đối với mỗi từ khóa w, có một danh sách L_w sao cho mỗi phần tử trong danh sách L_w là một cặp $(index, ind)$ trong đó ind là identifier của tài liệu chứa từ khóa w, và index là một con trỏ đến phần tử phần trước (hoặc tiếp theo) trong L_w. Một truy vấn tìm kiếm cho từ khóa w có thể dễ dàng trả về bằng cách đưa ra $index$ của phần tử mới nhất trong L_w và giải mã từng $index$ của phần tử trước đó để khôi phục lại tất cả định danh. Tuy nhiên, nó không phù hợp với mục tiêu của forward search privacy, bởi vì khó có thể được một phần của các phần tử mà không rò rỉ chúng thuộc danh sách nào và mối quan hệ với các phần tử khác trong cùng một danh sách. Làm thế nào để đạt được mức độ bảo mật cao nhất trong khi vẫn duy trì hiệu quả của SSE vẫn là một thử thách lớn.

d. Những đóng góp của các tác giả

Giải thích giới hạn bảo mật hiện tại đối với forward privacy và đề xuất một khái niệm nâng cao cho forward search privacy. Đảm bảo rằng các tìm kiếm trên các tài liệu mới được thêm vào không làm rò rỉ thông tin truy vấn trước đây. Forward search private SSE sẽ rò rỉ ít thông tin hơn SSE chỉ đáp ứng khái niệm forward privacy ban đầu. Miêu tả ứng dụng của FsP trong việc xây dựng các ứng dụng mã hóa an toàn và nâng cao hiệu quả trong việc thiết kế các cơ sở dữ liệu được mã hóa.

Thiết kế lược đồ Khons để đạt được FsP và hỗ trợ truy vấn song song với tính bảo mật cao và hiệu quả.

e. Định nghĩa về quyền riêng tư

Các tác giả nhắc lại định nghĩa về forward privacy và backward privacy từ đó đưa ra một định nghĩa mới forward search privacy.

Sự lặp lại của token (tức là các từ khóa truy vấn) gửi đến server sẽ bị rò rỉ trong hầu hết các lược đồ SSE. Nếu sự rò rỉ này giới hạn ở truy vấn tìm kiếm, được gọi là search pattern. Nếu sự rò rỉ này bao gồm sự lặp lại của các từ khóa được cập nhật, được gọi là query pattern.

Một hàm rò rỉ dữ liệu $leak\ function\ \mathcal{L}$ sẽ giữ nguyên trạng thái của danh sách truy vấn query list Q.

Danh sách truy vấn query list Q: danh sách tất cả các truy vấn từ trước đến nay, và các đối tượng trong danh sách là (i, w) cho một truy vấn tìm kiếm trên từ khóa w, hoặc (i, op, in) cho một truy vấn cập nhật op với đầu vào là in. Số nguyên i là một dấu thời gian, ban đầu là 0 và được tăng lên ở mỗi truy vấn. Gọi sp (x) và qp (x) lần lượt biểu thị các mẫu tìm kiếm và truy vấn được định nghĩa là:

- $sp(x) = \{ j : (j, x) \in Q \}$ chỉ khớp với các truy vấn tìm kiếm

- $qp(x) = \{ j : (j, x) \in Q \}$ hoặc $(j, op, in) \in Q$ và x xuất hiện trong in

Các tác giả ký hiệu:

- TimeDB(w) là danh sách tất cả các tài liệu phù hợp với từ khóa w, ngoại trừ những tài liệu đã bị xóa, cùng với dấu thời gian của thời điểm chúng được chèn vào cơ sở dữ liệu.

- Update (w) là danh sách các dấu thời gian cập nhật trên w.

- Delhist(w) là danh sách các dấu thời gian cho tất cả các thao tác xóa, cùng với dấu thời gian của các mục được thêm vào đã bị xóa.

Forward Update Privacy

Định nghĩa về forward privacy truyền thống là máy chủ không thể tìm hiểu xem các tài liệu mới được cập nhất có giống với các từ khóa đã tìm kiếm trước đó hay không.

Definition Forward update privacy: Một lược đồ $\mathcal{L} - adaptively - secure$ SSE là một forward $-$ update $-$ private nếu update leakage function $\mathcal{L}^{Update}$ có thể được viết:

$$\mathcal{L}^{Update}(op, ind, W) = \mathcal{L}'(ind, |W|),$$

Trong đó ind biểu thị là định danh của tài liệu vừa được thêm vào, $|W|$ biểu thị số lượng từ khóa của tài liệu mới được thêm vào, $\mathcal{L}'$ là stateless.

Từ định nghĩa trên forward update privacy yêu cầu thông tin bị rò rỉ trong phép toán tìm kiếm không được nhiều hơn định danh tài liệu và số lượng từ khóa của tài liệu mới được thêm vào.

Forward Search Privacy

Trong lược đồ SSE, các mã thông báo tìm kiếm (search token) rò rỉ một lượng thông tin đáng kể. Điều này được ghi lại bởi $leakage\ function$:

$\mathcal{L}^{Search}(w) = \mathcal{L}'\big(TimeDB(w)\big)$ trong đó $\mathcal{L}'$ là stateless

FsP được xác định trên cơ sở FuP, nó còn ngăn máy chủ biết liệu tìm kiếm trên các tài liệu mới cập nhật có khớp với từ khóa đã tìm kiếm trước đó hay không. Lần đầu tiên các tác giả đưa ra khái niệm về strong forward privacy. Một lược đồ SSE đáp ứng strong forward search privacy nếu mã tìm không không bị rò rỉ thông tin. Các tác giả định nghĩa như sau:

Definition Strong forward update privacy:

Một lược đồ $\mathcal{L} - adaptively - secure$ SSE là một $strong\ forward - search - private$ nếu $function\ \mathcal{L}^{Search}$ có thể được viết:

$\mathcal{L}^{Search}(w) = \mathcal{L}'(\perp)$ trong đó $\mathcal{L}'$ là stateless

Đây là một khái niệm mạnh mẽ nhưng cũng rất khó để đạt được. Thực tế, có nghĩa là hoạt đồng tìm kiếm gần như không thực hiện được trừ khi sử dụng các giao thức ORAM hoặc PIR. Để đáp ứng thực tế, các tác giả đã xác định một khái niệm yếu hơn về forward search privacy chỉ rò rỉ một phần pattern.

Definition Weak forward update privacy: Đặt $S_w = \{w_1, .., w_x\}$ biểu thị một tập sub-keyword cho một keyword w, trong đó x là một hằng số. Một lược đồ $\mathcal{L} - adaptively - secure$ SSE là một $weak\ forward - search - private$ nếu $leakage\ function\ \mathcal{L}^{Search}$ được viết:

$$\mathcal{L}^{Search}(w_i) = \mathcal{L}'(TimeDB(w_i))$$

Trong đó $\mathcal{L}'$ là stateless và $|TimeDB(w_i)| = a_w$ với a_w là một hằng số.

Backward privacy

Một lược đồ SSE đáp ứng backward privacy nếu sau khi xóa tài liệu có định danh ind tương ứng với từ khóa w, máy chủ không thể hiển thị tài liệu đã xóa có định danh ind từ lần tìm kiếm tiếp theo của từ khóa w.

Definition $(\boldsymbol{BP-I})$: Một lược đồ $\mathcal{L}-adaptively-secure$ SSE là $insertion\ pattern\ revealing\ backward-private$ nếu $leakage\ function\ \mathcal{L}^{Search}$ được viết:

$$\mathcal{L}^{Update}(o, w, ind) = \mathcal{L}'(op),$$

$$\mathcal{L}^{Search}(w) = \mathcal{L}''(TimeDB(w)),$$

Trong đó $\mathcal{L}'$ và $\mathcal{L}''$ là stateless và $|TimeDB(w_i)| = a_w$ với a_w là một hằng số.

Definition $(\boldsymbol{BP-II})$: Một lược đồ $\mathcal{L}-adaptively-secure$ SSE là $update\ pattern\ revealing\ backward-private$ nếu $leakage\ function\ \mathcal{L}^{Search}$ được viết:

$$\mathcal{L}^{Update}(o, w, ind) = \mathcal{L}'(op, w),$$

$$\mathcal{L}^{Search}(w) = \mathcal{L}''(TimeDB(w), Updates(w)),$$

Trong đó $\mathcal{L}'$ và $\mathcal{L}''$ là stateless và $|TimeDB(w_i)| = a_w$ với a_w là một hằng số.

Definition $(\boldsymbol{BP-III})$: Một lược đồ $\mathcal{L}-adaptively-secure$ SSE là $weakly\ backward-private$ nếu $leakage\ function\ \mathcal{L}^{Search}$ được viết:

$$\mathcal{L}^{Update}(o, w, ind) = \mathcal{L}'(op, w),$$

$$\mathcal{L}^{Search}(w) = \mathcal{L}''(TimeDB(w), DelHist(w)),$$

Trong đó $\mathcal{L}'$ và $\mathcal{L}''$ là stateless và $|TimeDB(w_i)| = a_w$ với a_w là một hằng số.

f. Tổng quan về công nghệ

Ở phần này giới thiệu 02 công nghệ giúp đạt được forward privacy.

Partitioning Technique

Trong các lược đồ SSE, các index (chỉ mục) được sử dụng rộng rãi. Bảng inverted index được sử dụng để hỗ trợ các search queries truy vấn tìm kiếm (truy vấn tìm kiếm) dưới dạng một cặp (key, value). Trong đó:

- Key: là keyword (từ khóa)

- Value: là một danh sách các identifier document (định danh tài liệu) chứa từ khóa này này.

Với một từ khóa cho trước có thể trích xuất tất cả các tài liệu có chứa từ khóa một cách hiệu quả.

Content/ document	ind (identifier document)	Keyword
D1	1	w1, w2, w3
D2	2	w4, w5
D3	3	w1, w3, w4, w6
D4	4	w2, w3. w7
D5	5	w6, w8, w9, w10

Bảng 3.1: Bảng Document index

Keyword	ind
W1	1, 3
W2	1, 4
W3	1, 3, 4
W4	2, 3
W5	2
W6	3, 5
W7	4
W8	5
W9	5
W10	5

Bảng 3.2: Bảng Inverted index

Trong Partitioning Technique: Các tác giả chia bảng inverted index thành các disjoint partition (phân vùng riêng biệt) và tạo ra một sub-keyword (từ khóa phụ(cho mỗi phân vùng để giảm thiểu thông rò rỉ trong SSE.

Bằng cách này, một search token (mã tìm kiếm) của một từ khóa sẽ trở thành nhiều mã tìm kiếm, mỗi mã tìm kiếm cho các partition (phân vùng) khác nhau.

Cụ thể hơn, các tác giả thêm identifier (định danh) của tài liệu vào một phân vùng bằng cách sử dụng sub-keyword có nguồn gốc từ w làm khóa khi thêm một tài liệu có chứa từ khóa w. Khi thực hiện truy vấn tìm kiếm, tác giả cho phép người dùng gửi một mã tìm kiếm của một sub-keyword để tìm kiếm trên một sub-set of documents (tập con các tài liệu) trong phân vùng này. Nếu chỉ thiết lập duy nhất một phân vùng cho từ khóa, nó sẽ trở thành inverted index truyền thống.

Content/ document	ind	Keyword
D1	1	w1, w2, w3
D2	2	w4, w5
D3	3	w1, w3, w4, w6
D4	4	w2, w3. w7
D5	5	w6, w8, w9, w10

Bảng 3.3: Document index

Phân vùng được chia như sau:

- Phân vùng 1

Content/ document	ind	Keyword
D1	1	w1, w2, w3
D2	2	w4, w5
D3	3	w1, w3, w4, w6

Bảng 3.4: Document Index phân vùng 1

- Phân vùng 2

Content/ document	ind	Keyword
D4	4	w2, w3. w7
D5	5	w6, w8, w9, w10

Bảng 3.5: Document Index phân vùng 2

Bảng inverted index cho từng phân vùng:

Keyword	ind
W1,1	1, 3
W2,1	1
W3,1	1, 3
W4,1	2, 3
W5,1	2
W6,1	3
W7,1	
W8,1	
W9,1	
W10,1	

Bảng 3.6: Bảng inverted index tương ứng phân vùng 1

Keyword	ind
W1,2	
W2,2	4
W3,2	4
W4,2	
W5,2	
W6,2	5
W7,2	4
W8,2	5
W9,2	5
W10,2	5

Bảng 3.7: Bảng inverted index tương ứng phân vùng 2

Hidden Pointer Technique (HPT)

Để sử dụng partitioning technique trong SSE, cần phải xây dựng các danh sách mã hóa (encrypted list) để có thể lưu trữ tất cả các chỉ mục ở máy chủ một cách bảo mật. Đầu tiên xác định cấu trúc dữ liệu:

- Một data block bao gồm four-tuple (bộ 4 dữ liệu): id, data, key, ptr
    + id: block identifier
    + data: a piece of data
    + key: encryption key
    + ptr: identifier của another block (suffix block)
- Nếu một block không có suffix block, key được set thành ⊥

-Trong một block b: các trường data, key, ptr phải được mã hóa.

+ b.id được biểu thị là id của block b.

+b.values là tất cả các nội dung khác của b bao gồm: b.data, b.key, b.ptr

- Gọi L là danh sách các khối dữ liệu (list of data blocks).
    + Head block: block mới nhất được thêm vào L.
    + Tail block: block cũ nhất trong L.

HPT bao gồm các thuật toán sau:

Thuật toán 1: $\boldsymbol{AddHead}(\boldsymbol{L}, \boldsymbol{id}, \boldsymbol{value}, \ \mathbf{1}^{\boldsymbol{\lambda}})$: Thêm một block mới như một head block vào list L.

1: Tạo một data block $\mathrm{b} = (\mathrm{id}, \mathrm{data}, \mathrm{L.head.key}, \mathrm{L.head.id})$

2: Lấy mẫu một khóa ngẫu nhiên k từ $\{0,1\}^{\lambda}$

3: Dùng k để mã hóa giá trị k.value

4: Add b vào L

Hình 3.4: Thuật toán AddHead

Thuật toán 2: $\boldsymbol{RetrieveABlock}(\boldsymbol{L}, \boldsymbol{id}, \boldsymbol{k})$: Trích xuất một data block từ list L

1: Tìm block b bằng identifier id

2: Giải mã giá trị b.value bằng khóa k tương ứng

3: Trả về b

Hình 3.5: Thuật toán truy xuất block

Thuật toán 3: $\boldsymbol{RetrieveList}(\boldsymbol{L}, \boldsymbol{id}, \boldsymbol{k})$: trích xuất tất cả các data block từ sublist của R bằng cách gọi RetrieveABlock(L, id, k) lặp lại liên tục cho đến tail block ($\mathrm{b.key} = \perp$)

Hình 3.6: Thuật toán truy xuất toàn bộ danh sách

Xây dựng inverted index (w, L_w). L_w là một list sử dụng HPT. Client có thể giữ phần đầu của danh sách và lưu trữ Lw trên server. List L_w được update bởi client bằng cách thêm vào một block mới. Người dùng có thể tìm kiếm index bằng cách tiết lộ id của head block và encryption key cho server.

Xây dựng forward index sử dụng HTP. Ánh xạ document identifier đến danh sách từ khóa có trong tài liệu.

Ví dụ: Từ Bảng 3.6 và Bảng 3.7 xây dựng một danh sách với từ khóa từ w1 đến w3 trên phân vùng 1 và phân vùng 2 tương ứng như sau:

		id	data	key	ptr
$key_1 \rightarrow$	$L_{w_{1,1}}$	id_1	ind_1	null	null
$key_2 \rightarrow$		id_2	ind_3	key_1	id_1
	$L_{w_{1,2}}$	null	null	null	null
$key_3 \rightarrow$	$L_{w_{2,1}}$	id_3	ind_1	null	null
$key_4 \rightarrow$	$L_{w_{2,2}}$	id_4	ind_4	null	null
$key_5 \rightarrow$	$L_{w_{3,1}}$	id_5	ind_1	null	null
$key_6 \rightarrow$		id_6	ind_3	key_5	id_5
$key_7 \rightarrow$	$L_{w_{3,2}}$	id_7	ind_4	null	null

Hình 3.7: Minh họa về công nghệ HPT

g. KHONS: Lược đầu SSE đầu tiên với forward search privacy

A forward and backward secure SSE scheme: Khons. Đáp ứng weak forward search privacy và forward update privacy (vì vậy có thể mở rộng ra backward privacy (BP-II)).

Kết hợp inverted index và forward index để tìm kiếm, cập nhật tài liệu hiệu quả và quan trọng hơn là cung cấp xóa ngay lập tức.

Cấu trúc lưu trữ

Ngữ cảnh:

- Tập từ khóa $W = \{w_1, w_2, \dots, w_y\}$ được phân hoạch trong x phân vùng.

- Tập tài liệu $D = \{D_1, D_2, \dots, D_z\}$

- Tập các định danh của tài liệu $I_D = \{ind_1, ind_2, \dots, ind_v\}$

- Máy chủ lưu trữ các data block trong một dictionary $\mathbb{D}$, $\mathbb{D}[\mathrm{id}]$ lưu trữ một data block với định danh id.

- Mỗi từ khóa w lấy một tập các sub-keyword

$S_w = \{ w_i \mid E_{K_s}(\mathrm{w}, \mathrm{i}), 1 \leq \mathrm{i} \leq \mathrm{x}\}$, K_s là encryption key.

- Mỗi sub-keyword w_i xây dựng một danh sách L_{w_i} cho mỗi truy vấn đơn phân vùng được coi là một inverted index.

- Để hỗ trợ full query search (truy vấn đầy đủ), với mỗi từ w, các tác giả thực hiện một số thay đổi ở tail block của các danh sách. Hạn chế cũ: Mỗi lần truy vấn chỉ trả về tối đa số định danh ind của tài liệu liên quan đến từ khóa w_i trên một phân vùng, không thể thực hiện full query search, do ptr field trong tail block của list có giá trị null nên tìm kiếm sẽ giới hạn theo phân vùng vì không có pointer đến block của phân vùng khác.

- Để giải quyết hạn chế cũ: giả sử với từ khóa w_i có x phân vùng, kỳ vọng trả về tất cả các ind có chứa từ khóa w_i từ x phân vùng: $L_{w_{i,1}}$, $L_{w_{i,2}}$, … , $L_{w_{i,x}}$.Mã hóa id và encryption của head block trong list $L_{w_{i,j-1}}$ sau đó thêm các giá trị đã được mã tương ứng vào data field và key field của tail block trong list $L_{w_{i,j}}$

Ví dụ: Danh sách tương ứng sau khi đã đã thay đổi ở tail block:

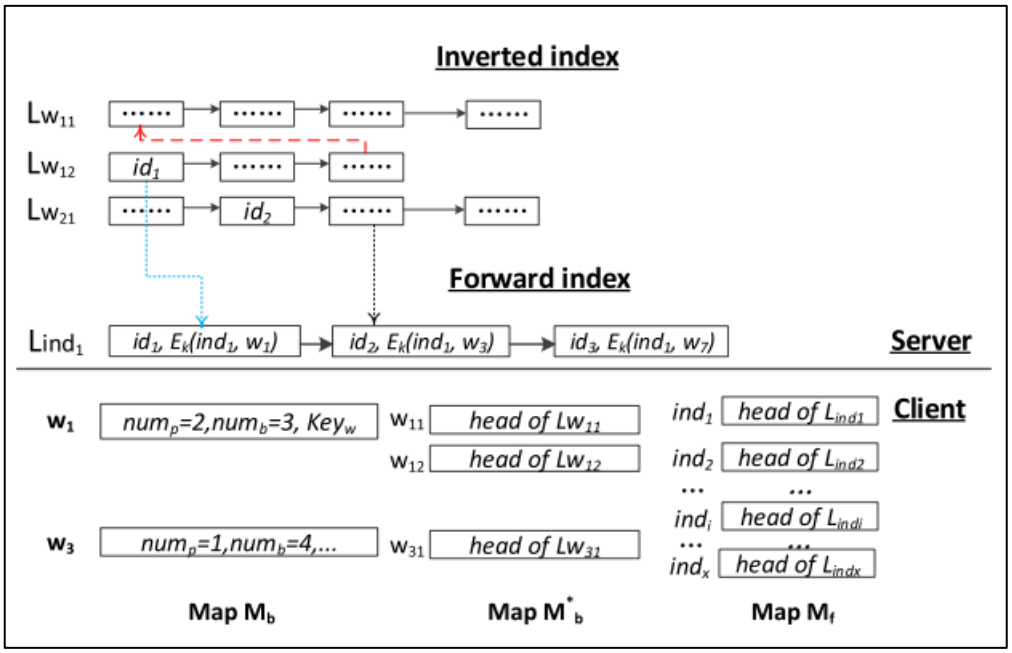
		id	data	key	ptr
$key_1 \rightarrow$	$L_{w_{1,1}}$	id_1	ind_1	null	null
$key_2 \rightarrow$		id_2	ind_3	key_1	id_1
	$L_{w_{1,2}}$	null	null	null	null
$key_3 \rightarrow$	$L_{w_{2,1}}$	id_3	ind_1	null	null
$key_4 \rightarrow$	$L_{w_{2,2}}$	id_4	$ind_4, \{id_3\}k_{2,2}$	$\{key_3\}k_{2,2}$	null
$key_5 \rightarrow$		id_5	ind_1	null	null

$key_6 \rightarrow$	$L_{w_{3,1}}$	id_6	ind_3	key_5	id_5
$key_7 \rightarrow$	$L_{w_{3,2}}$	id_7	$ind_4, \{id_6\}k_{7,2}$	$\{key_6\}k_{7,2}$	null

Hình 3.8: Minh họa về công nghệ HPT thay đổi ở tail block

- Mã hóa mang lại vấn đề là các key lưu trữ ở đâu? Do đó cần xây dựng một list L_w khác. Nó chứa một block cho mỗi L_{w_i} lưu trữ key để decrypt data lưu trữ trong tail block của L_{w_i}.

- Để hỗ trợ immediate delection (xóa ngay lập tức) đối với mỗi tài liệu có định danh ind. Sẽ có một list L_{ind} đóng vai trò là forward index.



Hình 3.9: Cấu trúc lưu trữ của thuật toán KHONS

- Client lưu trữ key K_s và map M_b , M_{b^*}, M_f . Key K_s là user secret key để tạo ra one-time key mã hóa dữ liệu. Map M_b , M_{b^*} lưu trữ trạng thái của từng keyword và sub-keyword. M_f lưu trữ trạng thái của từng tài liệu.

- Với mỗi keyword w_y

$M_b[w_y]$	num_p	1	tổng số phân vùng
	cnt_p	0	số lượng block
	key_w	$\perp$	encryption key của tail block trong latest (num_p-th) partition
	flag	false	thể hiện khi nào phân vùng mới nhất (nump-th) đã được truy cập

Bảng 3.8: Bảng trạng thái từ khóa

- Với mỗi sub-keyword

$M_{b^*}[w_{y.x}]$	id	$\perp$	identify của head block trong list $L_{w_{y,x}}$
	key	$\perp$	encryption key của head block trong list $L_{w_{y,x}}$
	cnt	0	số lượng block trong phân vùng $L_{w_{y,x}}$liên quan đến sub-keyword $w_{y.x}$
	flag	false	thể hiện khi partition của sub-keyword $w_{y.x}$ đã được truy cập

Bảng 3.9: Bảng trạng thái của từ khóa phụ [27]

- Với mỗi document ind, M_f $[ind]$ = (id,key) tức là thông tin con trỏ của khối đầu của danh sách Lind.

Như vậy về cấu trúc lưu trữ thuật toán đã hoàn thành.

h. Thuật toán KHONS

Người dùng khởi tạo ngẫu nhiên khóa người dùng K_s cùng các map $\mathbf{M}_b, \mathbf{M}_{b^*}, \mathbf{M}_f$ và dictionary $\mathbb{D}$.

Thuật toán 1: Khons**.Setup()**

1: $K_s \xleftarrow{\$} \{0,1\}^\lambda$, $\text{key}_h \xleftarrow{\$} \{0,1\}^\lambda$
2: $\mathbf{M}_b, \mathbf{M}_{b^*}, \mathbf{M}_f \leftarrow$ emptymap
3: $\mathbb{D} \leftarrow$ empty dictionary

Hình 3.10: Mã giả thuật toán 1- Khons.Setup()

Thêm các tài liệu cùng với định danh tài liệu và các từ khóa w tương ứng.

Thuật toán 2: Khons**.Update**$(\text{add}, w, \text{ind}, \sigma; \text{EDB})$

Client:

1: $(id_f, \text{key}_f) \leftarrow (\mathbf{M}_f[\text{ind}].\text{id}, \mathbf{M}_f[\text{ind}].\text{key})$

forward index

2: $id_f^* \xleftarrow{\$} \{0,1\}^\lambda, \text{key} \xleftarrow{\$} \{0,1\}^\lambda$
3: $\text{mask} \leftarrow H_1(\text{key}, id_f^*)$
4: $\text{value} \leftarrow E_{K_s}(\text{ind}||w)||\text{key}_f||id_f$
5: $(b.id, b.\text{value}) \leftarrow (id_f^*, \text{value} \oplus \text{mask})$
6: $\mathbf{M}_f[\text{ind}].id \leftarrow id_f^*, \mathbf{M}_f[\text{ind}].\text{key} \leftarrow \text{key}$

inverted index

7: $(\text{num}_p, \text{cnt}_p, \text{key}_w) \leftarrow (\mathbf{M}_b[w].\text{num}_p, \mathbf{M}_b[w].\text{cnt}_p, \mathbf{M}_b[w].\text{key}_w)$
8: $w_i \leftarrow H(\text{key}_h, w||\text{num}_p)$
9: $(id_b, \text{key}_b) \leftarrow (\mathbf{M}_{\boldsymbol{b}}^*[w_i].id, (\mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{key})$
10: $(\text{cnt}_p, \text{flag}_p) \leftarrow (\mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{cnt}, (\mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{flag})$
11: **IF** $(\text{cnt}_p = P \;||\; \text{flag}_b = \text{true})$
12: **IF** $(\text{flag}_b = \text{true})$
13: padding $P -$ cnt_p dummy blocks to w_i
14: $\text{num}_p \leftarrow \text{num}_p + 1,\ \text{cnt}_p \leftarrow 1$
15: $w_i \leftarrow H(\text{key}_h, w||\text{num}_p)$
16: initialize $\mathbf{M}_{\boldsymbol{b}}^*[w_i]$
17: **ELSE**
18: $\text{cnt}_p \leftarrow \text{cnt}_p + 1$
19: $id_b^* \xleftarrow{\$} \{0,1\}^\lambda, \text{key}^* \xleftarrow{\$} \{0,1\}^\lambda$

20: $\text{mask}_2 \leftarrow H_2(\text{key}^*, id_b^*)$

21: **IF** $(\text{cnt}_p = 0)$ // Add the first block into the list

22: $id_t \xleftarrow{\$} \{0,1\}^\lambda,\ \text{key}_t \xleftarrow{\$} \{0,1\}^\lambda$

23: $\text{mask}_3 \leftarrow H_3(\text{key}_t, w)$

24: $B_{\text{tail}}.id \leftarrow id_t$

25: $w_{i-1} \leftarrow H(\text{key}_h, w||\text{num}_{p-1})$

26: $B_{\text{tail}}.\text{value} \leftarrow (\text{key}_w||\mathbf{M}_{\boldsymbol{b}}^*[w_{i-1}].\text{key}||\mathbf{M}_{\boldsymbol{b}}^*[w_{i-1}].id) \oplus \text{mask}_3$

27: $id_b \leftarrow id_t, \text{key}_b \leftarrow \perp, \mathbf{M}_b[w].\text{key}_w \leftarrow \text{key}_t$

28: $(b^*.id, b^*.\text{value}) \leftarrow (id_b^*, (id_f^*||\text{key}_b||id_b) \oplus \text{mask}_2)$

29: $(\mathbf{M}_{\boldsymbol{b}}^*[w_i].id \leftarrow id_b^*, (\mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{key}) \leftarrow \text{key}^*, \mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{cnt}++$

30: Send block b, b^* and B_{tail} $(if\ exists)$ to the server .

Server:

31: $\mathbb{D}[b.id] = b.\text{value}$

32: $\mathbb{D}[b.id] = b^*.\text{value}$

33: **IF**(B_{tail} exists)

34: $\mathbb{D}[B_{\text{tail}}.id] = B_{\text{tail}}.\text{value}$

Hình 3.11: Mã giả thuật toán 2 - Khons.Update [27]

Thực hiện xóa các tài liệu cùng từ khóa w tương ứng.

Thuật toán 3: Khons**.Update**(delete, ind, σ; EDB)

Client:

1: $(id, \text{key}) \leftarrow (\mathbf{M}_f[\text{ind}].id, \mathbf{M}_f[\text{ind}].\text{key})$

2: Send $(\text{i}d, \text{key})$ to the server.

Server:

3: **REPEAT**

4: $b \leftarrow \mathbb{D}[id]$

5: Delete the block b from the dictionary $\mathbb{D}$

6: $\text{mask} \leftarrow H_1(\text{key}, id)$

7: $b.\text{value} \leftarrow b.\text{value} \oplus \text{mask}$

8: $(id, \text{key}) \leftarrow (b.id_f, \text{b.key}_f)$

9: **UNTIL** $(id = \perp)$

Hình 3.12: Mã giả thuật toán 3 - Khons.Update(delete)

Tìm kiếm các tài liệu tương ứng với từ khóa w

Thuật toán 4: Khons**.Search**$(w, i, \sigma;$ EDB)

Client:

$w_i \leftarrow H(\text{key}_h, w||i)$
$(id, \text{key}) \leftarrow (\mathbf{M}_b^*[w_i].id, \mathbf{M}_b^*[w_i].\text{key})$
$\mathbf{M}_b^*[w_i].\text{flag} \leftarrow \text{true}$
Send token (id, key) to the server.

Server:

$S \leftarrow$ empty set, $j \leftarrow 0$
REPEAT
    $b \leftarrow \mathbb{D}[id]$
    $\text{mask}_2 \leftarrow H_2(\text{key}, id)$
    $b.\text{value} \leftarrow b.\text{value} \oplus \text{mask}_2$
    $\mathbf{S} = \mathbf{S} \cup \mathbb{D}[b.id_f]$
    $(id, \text{key}) \leftarrow (b.id_b, b.\text{key})$
    IF (key $=\perp$)
    $id \leftarrow \perp$
UNTIL $(id = \perp)$
Send **S** to the Client

Client:

$\mathbf{S} \leftarrow \text{Decrypt}_{K_s}(\mathbf{S})$

Hình 3.13: Mã giả thuật toán 4 - Khons.Search()

Thuật toán 5 nâng cấp tìm kiếm, hỗ trợ tìm kiếm trên tất cả phân vùng.

Thuật toán 5: Khons**.Search**$(w, \sigma;$ EDB)

Client:

1: $(\text{key}_w, \text{num}_p) \leftarrow (\mathbf{M}_b[w].\text{key}, \mathbf{M}_b[w].\text{num}_p)$
2: $w_i \leftarrow H(\text{key}_h, w||\text{num}_p)$
3: $(id, \text{key}) \leftarrow (\mathbf{M}_{\boldsymbol{b}}^*[w_i].id, \mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{key})$
4: $\mathbf{M}_{\boldsymbol{b}}^*[w_i].\text{flag} \leftarrow \text{true}$
5: Send token (id, key) to the server.

Server:

6: $S \leftarrow$ empty set, $j \leftarrow 0$
7: **REPEAT**
8: $b \leftarrow \mathbb{D}[id]$
9: $\text{mask}_2 \leftarrow H_2(\text{key}, id)$
10: $b.\text{value} \leftarrow b.\text{value} \oplus \text{mask}_2$
11: $\mathbf{S} = \mathbf{S} \cup \mathbb{D}[b.id_f]$
12: $(id, \text{key}) \leftarrow (b.id_b, b.\text{key})$
13: **IF** (key $=\perp$)
14: **IF** ($\text{key}_w = \perp$) $id \leftarrow \perp$
15: **ELSE** $b^* \leftarrow \mathbb{D}[id]$
16 $\text{mask}_3 \leftarrow H_3(\text{key}_w, w)$
17: $b^*.\text{value} \leftarrow b^*.\text{value} \oplus \text{mask}_3$
18: $(id, \text{key}, \text{key}_w) \leftarrow (b^*.id, b^*.\text{key}, b^*.\text{key}_w)$
19: **UNTIL** $(id = \perp)$
20: Send **S** to the Client

Client:

21: $\mathbf{S} \leftarrow \text{Decrypt}_{K_s}(\mathbf{S})$

Hình 3.14. Mã giả thuật toán 5 - Khons.Search() nâng cấp

Tại phần 3.2, nhóm tác giả đã giới thiệu đầy đủ về thuật toán Khons với các hàm sử dụng trong nó.

CHƯƠNG 4. **HIỆN THỰC ĐỀ TÀI**

4.1. Mô hình hệ thống

Như đã trình bày ở phần báo cáo vấn đề, mục tiêu hướng đến của đề tài là tạo một cơ sở dữ liệu mã hóa có thể lưu an toàn trên cơ sở dữ liệu đám mây. Vì vậy, hệ thống bao gồm bốn thành phần được thiết kế là bốn server được mô tả như sau:

- Chủ sở hữu dữ liệu (DO): Là server sở hữu dữ liệu, bao gồm bản rõ của dữ liệu, thông tin nhân viên trong hệ thống với thuộc tính của họ.

- Trung tâm xác thực thuộc tính (AA): thực hiện chức năng tạo khóa cho người dùng dựa trên tập thuộc tính do DO cung cấp, thực hiện mã hóa dữ liệu theo từng chính sách riêng biệt với từng dữ liệu khác nhau.

- Cơ sở dữ liệu đám mây (CLOUD): nơi lưu trữ dữ liệu được mã hóa, cung cấp chức năng giải mã các dữ liệu dựa vào khóa do người dùng cấp.

- Người sử dụng dữ liệu (DU): sử dụng khóa do AA cung cấp, truy cập tại CLOUD và nhận về bản rõ dữ liệu.

Theo phần lý thuyết đã trình bày, có hai phương pháp nổi trội dành cho mã hóa dựa trên thuộc tính là KP-ABE và CP-ABE. Dựa theo những ưu điểm, nhược điểm của hai lược đồ này, nhóm sử dụng CP-ABE cho phương pháp hiện thực.

4.2. Ngữ cảnh hệ thống

Trong các ứng dụng của kiểm soát truy cập sử dụng mã hóa dựa trên thuộc tính, nhóm tác giả lựa chọn minh họa sử dụng một kịch bản chăm sóc sức khỏe [30, 31]. Bệnh viện giữ vai trò là DO hỗ trợ kiểm soát truy cập mịn dựa trên thuộc tính vào hồ sơ sức khỏe điện tử (EHR) [32, 33] bằng cách mã hóa và cung cấp hồ sơ đã được mã hóa cho nhân viên (DU). Người sử dụng hệ thống này điển hình bao gồm các nhân viên như lễ tân, thu ngân, bác sĩ, y tá, dược sĩ, quản trị hệ thống và người không phải là nhân viên bệnh viện như là bệnh nhân. Tài liệu EHR được chia thành các tài liệu nhỏ hơn như thông tin thanh toán (BillingInfo), thông tin liên hệ (ContactInfo), danh mục thuốc (Medication), vật lý trị liệu (PhysicalExam), chẩn đoán hình ảnh (LabReport) và có thể gồm nhiều thứ khác nữa.

Các chính sách của bệnh viện chỉ định người dùng nào có thể truy cập vào loại tài nguyên nào. Ví dụ một nhân viên thu ngân không cần có quyền truy cập vào các tài nguyên khác của EHR ngoại trừ thông tin thanh toán trong khi bác sĩ, y tá thì không cần phải có quyền truy cập thông tin này. Ví dụ như chính sách quy định "thông tin về bệnh nhân ung thư chỉ có thể được tiếp cận bởi bác sĩ điều trị chính của bệnh nhân". Ngoài ra,

các chính sách này có thể hỗ trợ một số bệnh nhân có nhu cầu bảo mật tối đa, ngoài các chính sách của bệnh viện. Ví dụ bệnh nhân có thể chỉ định chính sách riêng của cô ấy là: "chỉ có các bác sĩ và y tá có hỗ trợ chương trình bảo hiểm của cô ấy mới có thể xem EHR của cô". Hình 4.1 mô tả thuộc tính của nhân viên trong bệnh viện với các thuộc tính là các cột.

4.3. Hiện thực hệ thống

Phần này, dựa trên những kiến thức đã trình bày cũng như ngữ cảnh hệ thống, nhóm thực hiện một phần mềm hỗ trợ chức năng mã hóa và giải mã dựa trên thuộc tính của nhân viên trong bệnh viện đối với hồ sơ sức khỏe của bệnh nhân

empid	role	level	insurance1	insurance2	insurance3	insurance4
1	doctor	senior	MedC	ACME	MedA	MedB
2	cashier	senior	MedB	NULL	ACME	MedC
3	nurse	senior	MedC	ACME	MedA	MedB
4	doctor	junior	MedA	MedC	NULL	MedB
5	nurse	senior	MedC	MedA	NULL	MedB
6	nurse	junior	MedA	MedC	NULL	ACME
7	doctor	junior	ACME	MedB	MedC	MedA
8	doctor	senior	MedB	ACME	NULL	MedC
9	doctor	junior	MedB	MedA	ACME	MedC
10	doctor	senior	MedB	MedA	NULL	ACME
11	nurse	senior	ACME	MedB	MedC	MedA
12	nurse	junior	MedC	MedB	NULL	ACME
13	cashier	junior	NULL	MedB	MedC	ACME
14	nurse	junior	NULL	ACME	MedB	MedC
15	cashier	junior	MedB	MedA	NULL	MedC
16	doctor	junior	NULL	MedA	ACME	MedB
17	nurse	senior	MedA	NULL	MedB	ACME
18	nurse	senior	MedB	NULL	MedA	ACME
19	doctor	junior	ACME	NULL	MedB	MedC
20	cashier	junior	NULL	MedB	MedC	ACME

Hình 4.1: Thuộc tính của các nhân viên

Dữ liệu về bệnh nhân của bệnh viện được biểu diễn tóm tắt như trong Hình 4.2.

id	name	phone	email	sex	bill	contact	insurance
1	Laural Banthorpe	534-659-8814	lbanthorpe0@1688.com	0	41.03	1 Kennedy Way	ACME
2	Tori Bartell	318-451-3168	tbartell1@nymag.com	1	58.25	9 Sommers Parkway	ACME
3	Emili Tovey	659-882-7007	etovey2@salon.com	0	28.37	139 Dryden Road	ACME
4	Jaye Binford	574-911-8573	jbinford3@chicagotribune.com	1	43.44	7195 Roth Place	ACME
5	Abbe Allmond	227-499-9730	aallmond4@parallels.com	1	30.09	30286 Loftsgordon Lane	ACME
6	Lani Harflete	489-699-2352	lharflete5@domainmarket.com	1	38.01	1 Warbler Point	MedB
7	Emmet Blethyn	162-604-9329	eblethyn6@ucoz.com	0	75.56	63 Browning Avenue	MedC
8	Chilton Allsobrook	220-921-0332	callsobrook7@state.tx.us	1	14.94	74063 Fremont Center	MedB
9	Eugenio Merdew	304-844-4847	emerdew8@cafepress.com	0	48.98	75 Fair Oaks Junction	ACME
10	Dalton Coltart	132-475-3375	dcoltart9@ucla.edu	1	97.37	5003 Kenwood Park	MedB
11	Brad Denkel	807-990-8869	bdenkela@tripadvisor.com	1	43.44	3 Anderson Road	MedB
15	Minor Cossam	472-818-1850	mcossame@nydailynews.com	1	44.73	3799 Northfield Trail	MedA
16	Huey Yurlov	450-996-2550	hyurlovf@nytimes.com	0	55.31	5641 Glendale Place	MedA
17	Katherina Shead	389-595-6588	ksheadg@sbwire.com	1	57.95	57 Surrey Trail	MedC
18	Chick Mewes	429-352-4384	cmewesh@tripod.com	0	12.21	357 Old Shore Plaza	MedB
19	Anatol Crichten	876-260-1034	acrichteni@gnu.org	0	47.44	25 Cardinal Avenue	MedB
20	Lindsay Hadeke	251-643-3867	lhadekej@sfgate.com	0	7.85	92484 Dovetail Center	MedC
21	Katti Spini	965-180-9505	kspinik@smh.com.au	0	74.5	5 Del Mar Street	MedC
22	Alberto Sarsfield	364-984-4448	asarsfieldl@nature.com	0	32.56	94086 Iowa Circle	MedB

Hình 4.2: Dữ liệu của bệnh nhân

Giả sử trong hệ thống này, có các chính sách cơ bản được quy định bằng thuật ngữ như sau

- Những thông tin về hóa đơn và địa chỉ liên hệ chỉ có nhân viên thu ngân được phép xem.

- Những thông tin về số điện thoại, thư điện tử và chương trình chăm sóc sức khỏe của bệnh nhân thì cả bác sĩ, y tá và thu ngân đều xem được.

Ngoài ra, các bệnh nhân thuộc chương trình bảo hiểm ACME có riêng quy định rằng tất cả các thông tin của họ chỉ được xem bởi những nhân viên là thành viên của chương trình bảo hiểm ACME. Vậy, hệ thống chúng ta cần phải mã hóa các thông tin dựa theo ba chính sách đã đề cập.

Sau khi khởi tạo các tham số cho hệ thống, AA xuất ra các tham số này và lưu tại server, các thuật toán sử dụng sau này sử dụng các tham số này để tính toán.

```
root@AA:/var/www/html/project-c++/param# cat secparam.txt
AAAAFqpvyYPtNX3RDwX3lIWpP1AOE3Ztc2sAAAB3oQVhbHBoYaEjsQAgBSafamOxk3FiCqQIhY2abobsR8CWkvnsRX6QX7jMJ52hA2cyYaFEs6FBAx13g2E1V
LRY8E0+SSibZ5pSrNs/KpuQTCZw5izuI6Z+D6wr4xeBHX8VtcfSQz4f5ynuC/vJZAc+SukwNMfPUhc=root@AA:/var/www/html/project-c++/param#
root@AA:/var/www/html/project-c++/param# cat pubparam.txt
AAAAFqpvybTTESjviwDUzKXCk9WsWZRtcGsAAAHToQFBsgEEtLIBAAMKul/8X9Sn+m5mxBNQcikhTb3SOz2dXQFItYLUM8hKGXrD6KVEzCn1G1nxUSyaPEhcF
kqNt3dCuHwobRhqomwd8HOJgvxWGhh5ktLc/Efq1PS9y4KzY22F5SuNzR2c+gmJmzsJeQ75PKceaoERjJWCeNlGzpXXK430Ns3YA7daHPcKax/7EvDHklVrwm
0MlV+pR9dK/v0bZy7hmrrpo0EEanNm2GEfSMpxjqjB/GLwBB8KYWXQYYoD+1gZZZHw/ArfKBTxAWuK7aO0DS7IeRVjOhSNbKWv7SxKy99IFB82EE3YyA/2Sf2
p1tBcEcIatEUZnpZ4JbRNki6joNzTNg2hAmcxoSSyoSECBVsFSvjMtJdSOhGgZKUbllW42f+JRglNVCWgWMYmQzuhA2cxYaEksqEhAxi3G4AxUYb0Roy1Cpu9
kGKItxgS52qpBxKrk87TuwsEoQJnMqFEs6FBAhJy7rvINjYJlysR1mjflSF56yzS2ZTnoZRzUYkWx5LFEpN1QX3VWe/9K7DKgm2bO9NsXaEDDXzJDETtMy0Z3
i2hAWuhJR0AAAAg7vK+8jBCYQA586ngCSrbq96YP+Q2CewRHrSQPYmvZgo=root@AA:/var/www/html/project-c++/param# ls
pubparam.txt  secparam.txt
```

Hình 4.3: Các tham số công khai và bí mật sử dụng trong hệ thống

Sau đó, đối với mỗi người dùng có các thuộc tính khác nhau, AA tạo các khóa và cung cấp cho người dùng. Các thuộc tính này do chủ sở hữu dữ liệu cung cấp, dựa trên cơ sở dữ liệu về nhân viên của họ. Hình 4.4 mô tả một đoạn xử lý trong chương trình mô tả các tạo ra khóa của người dùng.

```
InitializeOpenABE();
OpenABECryptoContext cpabe("CP-ABE");
cpabe.importPublicParams(read_ob("param/pubparam.txt"));
cpabe.importSecretParams(read_ob("param/secparam.txt"));
string id = argv[1];
string listatt = shellcall("./getattlist " + id);
cpabe.keygen(listatt, "key");
string skBlob;
cpabe.exportUserKey("key", skBlob);
cout << id << " " << skBlob << endl;

ShutdownOpenABE();

return 0;
```

Hình 4.4: Đoạn xử lý tạo khóa cho người dùng

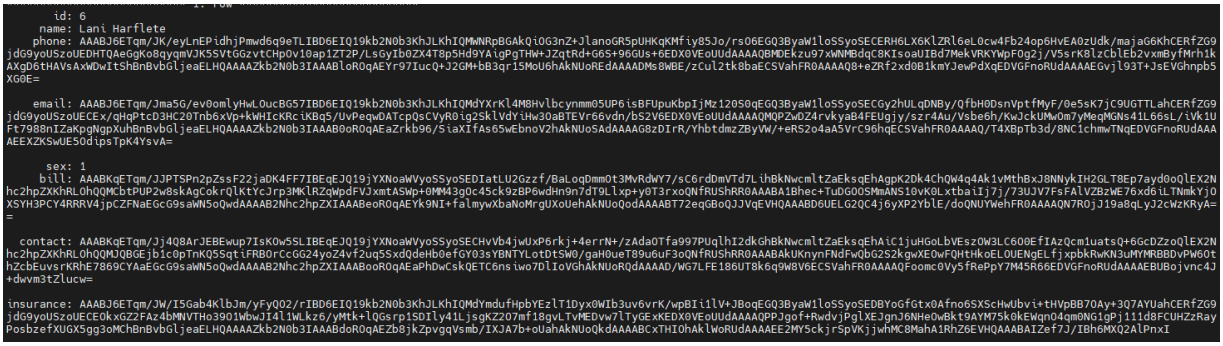
Việc tạo khóa này ngoài tham số chính của hệ thống và tập thuộc tính của người dùng, còn có các tham số ngẫu nhiên. Cùng một tập thuộc tính nhưng mỗi lần tạo khóa đều cho các khóa khác nhau. Điều này ngăn chặn được các cuộc tấn công thông đồng giữa những người dùng nhằm phá hoại hệ thống. Hình 4.5 cho thấy, cùng với số thứ tự nhân viên 13, các khóa cho ra khác nhau với cùng một tập thuộc tính.

```
root@AA:/var/www/html/project-c++# ./keygen 13
13 AAAAFqpvySI79HtVO5nDpmGcgIL9MT9rZXkAAAGqoQFLoUSzoUEDHq674yI1tYNrmHZNDFfOb7MH9XdkQAroHZmxtehfW78G6jWX0ssQ40oMoBuLTu8kk2
tDdx7Je2GBXFqUeLhsx6EHS1hfQUNNRaEksqEhAhj29SgFFqT3Gfctu2coZ0XojJGubYUn1EQRbm99VcQ9oQdLWF9NZWRCoSSyoSEDIkIWsBrOKlfnPsHfoQb
YRJqGRCpTXRnJAyx5nqPaHAehB0tYX01lZEOhJLKhIQMONMAtvad8akLekFyYv+kHBtKuUEgeSUiBzwmcIZcmNaEKS1hfY2FzaGllcqEksqEhAgdbVvQzKbCX
Ml0wWPSlq7FiuYX5ob2XWXrnGmGp0LdzoQlLWF9qdW5pb3KhJLKhIQMG65tOn+d+1Znu6LvJ3s6c7hcLdIy11QV1x9/YkJaXcKEBTKFEs6FBAyH9tSxhbfBc1
JehBy3uTUC6OsZl6ptcULx5IG1M5W26BIeboy3lbWW1k47dSqWihrhADIBBzElrAo0umRiksaWhBWlucHV0oR98Y2FzaGllcnxqdW5pb3J8TWVkQnxNZWRDfE
FDTUV8
root@AA:/var/www/html/project-c++# ./keygen 13
13 AAAAFqpvyRuXX8yY6g4ZdfUGn4uw5U5rZXkAAAGqoQFLoUSzoUECEUae/y/8kc4D7Av2bGf+zmd7q0+PNIArZggVxXhFgrQgnqS1FF/U+jcg+52ADvt5jO
rA9s2IElzcvWh5drzpwaEHS1hfQUNNRaEksqEhAwugwC89cnBWHM4zII/LCL7BC/83IKiKCxuR3anBdYW0oQdLWF9NZWRCoSSyoSECG+McMPxY8uWbcZ/VoQx
4gNcl9ugVB5jdKU7z6EGSGz6hB0tYX01lZEOhJLKhIQIFRsI7syWwZjCN276qyoEcFG9vhbL+m/tzxV825krhiqEKS1hfY2FzaGllcqEksqEhAg0Ile3iTn33
F5MjsyLELXVi75ZF9eZSqe72WVITBtxgoQlLWF9qdW5pb3KhJLKhIQIR5QAbMBHRj9kZ8khf3fs8RRX7ZIA5waieUkPIq1Yg76EBTKFEs6FBAgc504+ZOEXPS
Y9eIqdhWryEOJdxEpsWNjaQFPTMshzBBvGb7hAUkOv+ZgwyQO3H7r90XGNSuTrUlQ9Su8seTe2hBWlucHV0oR98Y2FzaGllcnxqdW5pb3J8TWVkQnxNZWRDfE
FDTUV8
```

Hình 4.5: Tạo khóa cho người dùng

Tại bước mã hóa, AA thực hiện mã hóa theo chính sách truy cập được quy định, sau đó lưu bản mã tại máy chủ đám mây. Hình 4.6 và Hình 4.7 cho thấy dữ liệu được lưu trên máy chủ. Bệnh nhân có định danh thứ tự là 2 thuộc chương trình ACME, vì vậy, toàn bộ thông tin của người này được mã hóa. Bệnh nhân số 6 thì không, vì vậy thông tin về tên và giới tính vẫn được thể hiện giống như bản rõ.

Hình 4.6: Dữ liệu mã hóa được lưu trên server (1)



Hình 4.7: Dữ liệu mã hóa được lưu trên server (2)

Ở phần giải mã, người dùng sẽ cung cấp khóa của họ cho máy chủ cùng giá trị bản mã muốn giải mã, máy chủ thực hiện giải mã và trả về bản rõ cho người dùng. Hình 4.8 thể hiện một phần trong tính năng giải mã, với việc cung cấp khóa và bản mã. Kết quả giải mã được biểu thị trong Hình 4.9.

```
InitializeOpenABE();
OpenABECryptoContext cpabe("CP-ABE");
cpabe.importPublicParams(read_ob("param/pubparam.txt"));
cpabe.importSecretParams(read_ob("param/secparam.txt"));
string pt1,testexpkey, ct;

testexpkey = "AAAAFqpvyUkkOhgdrBUKXqeJ8JZ0F2JrZXkAAAGooQFLoUSzoUECIV6Wuc0W/RdlyK5f+ak4uz749Yo
string cipher = "AAACO6ETqm/Jd3PojX/zgaNroPsKli2hNrICI6EJQ19jYXNoaWVyoSSyoSECGY9ER9oulKWC5Cdw
cpabe.importUserKey("key1", testexpkey);

cpabe.decrypt("key1", cipher, pt1);
cout << pt1 << endl;
ShutdownOpenABE();
```

Hình 4.8: Giải mã

Hình 4.9: Kết quả giải mã

Trong giới hạn về thời gian cũng như khả năng của đề tài, nhóm tác giả chỉ hiện thực một tính năng minh họa của hệ thống, chưa hiện thực thành sản phẩm có thể sử dụng trong thực tế. Chức năng tìm kiếm trên dữ liệu mã hóa cũng chưa thể thực hiện. Các chức năng hiện thực đầy đủ có triển vọng áp dụng trong môi trường thực tế tại các bệnh viện, trường học.

4.4. Nhận xét, đánh giá

Qua quá trình thực nghiệm thực tế trên hệ quản trị cơ sở dữ liệu MySQL, nhóm tác giả nhận thấy, CP-ABE phù hợp với mục đích tăng tính bảo mật trên cơ sở dữ liệu đặt ở đám mây. CP-ABE hỗ trợ truy cập mịn tới từng ô của bảng, có thể áp dụng cho các chính sách truy cập riêng lẻ, mang tính cá nhân. Thư viện OpenABE áp dụng cung cấp hàm hỗ trợ xóa các khóa của người dùng khi họ không còn thuộc tính phù hợp với chính sách truy cập.

Tuy nhiên, CP-ABE có một số đặc điểm không thuận lợi. Vì nhúng trực tiếp chính sách truy cập vào bản mã, dung lượng bản mã tăng lên nếu chính sách truy cập vào nó càng phức tạp. Vì vậy, trước khi áp dụng chính sách vào bản mã, cần thực hiện rút gọn nó thành dạng chuẩn tắc, tránh tình trạng chính sách chưa là tối giản.

Tốc độ xử lý của thư viện trên máy ảo linux không cao, việc sử dụng c++ đã đẩy nhanh quá trình thực hiện giải mã. Đối với những chính sách truy cập theo như phần hiện thực đã sử dụng, thời gian thực hiện luôn nhỏ hơn 0.01s đối với các bản mã. Thời gian giải mã không tăng tuyến tính theo độ dài bản mã mà có sự tăng giảm không đều, các lần giải mã cùng bản mã có biên độ thời gian thực hiện giữa giá trị cao nhất và giá trị nhỏ nhất khá lớn. Nhóm tác giả không đưa ra kết luận về tốc độ xử lý của thuật toán, thư viện.

Tùy theo mục đích sử dụng, ở ngữ cảnh này là đề cao tính bảo mật của tài liệu nhạy cảm, nhóm tác giả nhận định CP-ABE phù hợp và có thể triển khai thực tế. Quá trình xây dựng ứng dụng cần thiết kế rõ về cấu trúc của cơ sở dữ liệu, phân bố chính sách hiệu quả, không mã hóa toàn bộ hoặc dùng các chính sách truy cập chưa hoàn thiện cho toàn bộ cơ sở dữ liệu.

CHƯƠNG 5. **KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN**

5.1. Các kết quả đạt được từ luận văn

Như vậy, các nghiên cứu về kiểm soát truy cập thuộc tính trên dữ liệu mã hóa hỗ trợ truy vấn đã thành công cũng như hiện thực được một mô hình kiểm soát truy cập dựa trên thuộc tính. Nhóm tác giả xin tổng kết, đánh giá lại toàn bộ những điều mà nhóm đã đạt được trong quá trình làm luận văn:

- Đầu tiên về kiến thức và kinh nghiệm tích lũy được, mục tiêu nghiên của nhóm là nghiên cứu về Mã hóa và kiểm soát quyền truy cập trên hệ quản trị cơ sở dữ liệu SQL nhằm nâng cao quyền riêng tư tài liệu của người dùng trước lưu trữ đám mây từ bên cung cấp dịch vụ thứ ba không đáng tin cậy. Đây là một đề tài khá mới mẻ với nhóm, do đó mọi kiến thức thu được trong quá trình nghiên cứu là rất cần thiết cho những nghiên cứu liên quan trong công việc sau này.

- Thứ hai, nhóm đã nắm bắt được các thuật toán, các khái niệm về toán học ứng dụng mã hóa. Các ưu, nhược điểm của phương pháp quản lý truy cập dựa trên thuộc tính, phương pháp truy vấn trên dữ liệu mã hóa. Từ đó hiện thực và đề xuất một số giải pháp hiệu quả, có khả năng ứng dụng.

- Thứ ba, nhóm đã hiện thực được sơ bộ những tính năng mà hệ thống quản lý truy cập dựa trên thuộc tính cần có, đề ra được hướng phát triển của đề tài cũng như đánh giá tính khả thi của hệ thống trong môi trường thực tế.

5.2. Đề xuất và hướng phát triển

- Tiếp tục thực hiện hiện thực hoàn thiện các chức năng của hệ thống quản lý truy cập dựa trên thuộc tính, có khả năng ứng dụng thực tế thông qua các ứng dụng tiếp cận thân thiện với người dùng như website, ứng dụng điện thoại.

- Hiện thực phương pháp truy vấn trên dữ liệu mã hóa, kết hợp hỗ trợ truy vấn trên dữ liệu mã hóa đã có chức năng điều khiển truy cập.

TÀI LIỆU THAM KHẢO

[1] V. C. Hu *et al.*, "Guide to attribute based access control (ABAC) definition and considerations (draft)," *NIST special publication,* vol. 800, no. 162, 2013.

[2] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer, and W. Karl, "Scientific cloud computing: Early definition and experience," in *2008 10th ieee international conference on high performance computing and communications*, 2008: Ieee, pp. 825-830.

[3] V. Suming. "26 Cloud Computing Statistics, Facts & Trends for 2021." Cloudwards. https://www.cloudwards.net/cloud-computing-statistics/ (accessed 12-7, 2021).

[4] T. MORROW. "12 Risks, Threats, & Vulnerabilities in Moving to the Cloud." Software Engineering Institute - Carnegie Mellon University. https://insights.sei.cmu.edu/blog/12-risks-threats-vulnerabilities-in-moving-to-the-cloud/ (accessed 12-7, 2021).

[5] McAfee, "Cloud Adoption and Risk Report," 2020.

[6] J. Emms, "A definition of an access control systems language," *Computer standards & interfaces,* vol. 6, no. 4, pp. 443-454, 1987.

[7] R. Sandhu, E. Coyne, H. Feinstein, and C. Y. Role-Based, "Access Control Models," *IEEE computer,* vol. 29, no. 2, pp. 38-47, 2013.

[8] M. Nyanchama and S. Osborn, "Modeling mandatory access control in role-based security systems," in *Database Security IX*: Springer, 1996, pp. 129-144.

[9] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," MITRE CORP BEDFORD MA, 1973.

[10] S. Osborn, "Mandatory access control and role-based access control revisited," in *Proceedings of the second ACM workshop on Role-based access control*, 1997, pp. 31-40.

[11] Z. Tang, X. Ding, Y. Zhong, L. Yang, and K. Li, "A self-adaptive Bell–LaPadula model based on model training with historical access logs," *IEEE Transactions on Information Forensics and Security,* vol. 13, no. 8, pp. 2047-2061, 2018.

[12] J. McLean, "Security models and information flow," NAVAL RESEARCH LAB WASHINGTON DC CENTER FOR HIGH ASSURANCE COMPUTING SYSTEMS …, 1990.

[13] D. Zhu, Y. Yang, H. Jin, J. Shao, and W.-M. Feng, "Application of modified blp model on mobile web operating system," in *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016: IEEE, pp. 1818-1824.
[14] W. Ou, X. Wang, W. Han, and Y. Wang, "Research on trusted network model based on BLP model," in *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, 2009: IEEE, pp. 1137-1142.
[15] L. Yang, J. Wang, Z. Tang, and N. N. Xiong, "Using conditional random fields to optimize a self-adaptive bell-lapadula model in control systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.
[16] V. C. Hu, D. Ferraiolo, and D. R. Kuhn, *Assessment of access control systems*. Citeseer, 2006.
[17] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, "Attribute-based access control," *Computer*, vol. 48, no. 2, pp. 85-88, 2015.
[18] E. Yuan and J. Tong, "Attributed based access control (ABAC) for web services," in *IEEE International Conference on Web Services (ICWS'05)*, 2005: IEEE.
[19] N. S. Kumar, G. R. Lakshmi, and B. Balamurugan, "Enhanced attribute based encryption for cloud computing," *Procedia Computer Science*, vol. 46, pp. 689-696, 2015.
[20] M. Nabeel and E. Bertino, "Attribute Based Group Key Management," *Trans. Data Priv.*, vol. 7, no. 3, pp. 309-336, 2014.
[21] W. Dai *et al.*, "Implementation and evaluation of a lattice-based key-policy ABE scheme," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1169-1184, 2017.
[22] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*, 2007: IEEE, pp. 321-334.
[23] S. Moffat, M. Hammoudeh, and R. Hegarty, "A survey on ciphertext-policy attribute-based encryption (CP-ABE) approaches to data security on mobile devices and its application to IoT," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, 2017.
[24] M. I. Mihailescu and S. L. Nita, *Pro Cryptography and Cryptanalysis with C++ 20*. Apress, 2021.

[25] Y. Wang, J. Wang, and X. Chen, "Secure searchable encryption: a survey," *Journal of communications and information networks*, vol. 1, no. 4, pp. 52-65, 2016.
[26] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE symposium on security and privacy. S&P 2000*, 2000: IEEE, pp. 44-55.
[27] J. Li *et al.*, "Searchable symmetric encryption with forward search privacy," *IEEE Transactions on Dependable and Secure Computing*, 2019.
[28] S. Garg, P. Mohassel, and C. Papamanthou, "TWORAM: efficient oblivious RAM in two rounds with applications to searchable encryption," in *Annual International Cryptology Conference*, 2016: Springer, pp. 563-592.
[29] Y. Zhang, J. Katz, and C. Papamanthou, "All your queries are belong to us: The power of file-injection attacks on searchable encryption," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 707-720.
[30] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," *Journal of Computer Security*, vol. 18, no. 5, pp. 799-837, 2010.
[31] N. Shang, M. Nabeel, F. Paci, and E. Bertino, "A Privacy-Preserving Approach to Policy-Based Content Dissemination (Full Paper)."
[32] M. Eichelberg, T. Aden, J. Riesmeier, A. Dogac, and G. B. Laleci, "A survey and analysis of electronic healthcare record standards," *Acm Computing Surveys (Csur)*, vol. 37, no. 4, pp. 277-315, 2005.
[33] "XML in clinical researchand healthcare industries." http://xml.coverpages.org/healthcare.html (accessed 27-7, 2021).