BỘ KHOA HỌC VÀ CÔNG NGHỆ
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

HOÀNG NGỌC CẢNH

PHÁT TRIỂN MỘT SỐ PHƯƠNG PHÁP TRUY VẤN HIỆU QUẢ TRÊN CƠ SỞ DỮ LIỆU QUAN HỆ MÃ HOÁ

Ngành: Hệ thống thông tin
Mã số: 9.48.01.04

TÓM TẮT LUẬN ÁN TIẾN SĨ KỸ THUẬT

HÀ NỘI - 2025

Công trình được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học:

GS.TS. NGUYỄN HIẾU MINH

TS. NGÔ ĐỨC THIỆN

Phản biện 1:

Phản biện 2:

Phản biện 3:

Luận án được bảo vệ trước Hội đồng chấm luận án cấp Học viện họp tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

vào hồi: **giờ 00', ngày tháng năm 2025**

Có thể tìm hiểu luận án tại:

1. **Thư viện Quốc gia Việt Nam**
2. **Thư viện Học viện Công nghệ Bưu chính Viễn thông**

MỞ ĐẦU

1. GIỚI THIỆU

Ngày nay, việc người sở hữu dữ liệu (DO) triển khai CSDL nói chung và CSDLQH nói riêng trên nền tảng hạ tầng thuê ngoài (CS) ngày càng phổ biến. Tuy nhiên một vấn đề được đặt ra là làm sao để CSDL thuê ngoài đảm bảo được an toàn, ngăn cấm việc khai thác thông tin bất hợp pháp của các tổ chức/cá nhân không có thẩm quyền, kể cả nhà cung cấp dịch vụ (DSP). Rõ ràng ngoài việc DO cần có thêm các điều khoản hợp đồng chặt chẽ hơn về an ninh dữ liệu với DSP và tăng cường các lớp bảo mật cho đường truyền, máy chủ, hệ điều hành, hệ quản trị CSDL thì việc xây dựng các biện pháp chủ động che giấu nội dung dữ liệu nhạy cảm (bằng phương pháp mã hóa) trong tất cả các khâu: lưu trữ, gửi nhận và khai thác dữ liệu sao cho thông tin chứa trong dữ liệu luôn an toàn trên máy chủ và đường truyền là vô cùng quan trọng. Khi dữ liệu mã hóa được lưu trên CSDL thuê ngoài, đồng nghĩa CSDL này sẽ không được tồn tại các khóa hoặc metadata để phục vụ quá trình mã hóa, giải mã hoặc truy xuất thông tin. Do đó mọi yêu cầu truy vấn trên dữ liệu rõ từ người dùng phải được chuyển đổi thành các lệnh truy vấn trên dữ liệu mã hóa và gửi tới CSDL thuê ngoài để thực thi. Tuy nhiên dữ liệu nhạy cảm (dạng số, ký tự, logic, …) sau khi mã hóa bởi các thuật toán mật mã tiêu chuẩn như AES, DES, … sẽ không còn giữ được các tính chất vốn có ban đầu như: thứ tự, so sánh, tính toán số học, thống kê,… Vì vậy bài toán truy vấn hiệu quả trên dữ liệu mã (đặc biệt là truy vấn trên CSDLQH mã hoá thuê ngoài) có tính cấp thiết và được quan tâm đặc biệt trong những năm gần đây.

2. MỤC TIÊU CỦA LUẬN ÁN

Mục tiêu chung: Đề xuất các mô hình, lược đồ và thuật toán truy vấn hiệu quả với hai dạng thức dữ liệu phổ biến nêu trên theo các chuỗi điều kiện truy vấn hay được sử dụng trong CSDLQH mã hoá thuê ngoài.

Mục tiêu cụ thể:

- Nghiên cứu áp dụng mô hình DAS-PROXY trong triển khai các lược đồ mã hoá đối xứng có thể tìm kiếm (SSE) dựa trên chỉ mục mù.

- Nghiên cứu các kiến thức liên quan về CSDL quan hệ, cấu trúc dữ liệu B+Tree, bộ lọc Bloom, … phục vụ xây dựng các chỉ mục mù và thuật toán truy vấn hiệu quả.

- Đề xuất các lược đồ SSE mới dựa vào chỉ mục mù cho phép truy vấn hiệu quả dữ liệu dạng số và ký tự trên CSDL quan hệ mã hóa thuê ngoài theo các điều kiện truy vấn thường được người dùng yêu cầu.

- Thực nghiệm và phân tích tính hiệu quả của các lược đồ SSE đã đề xuất, trong đó tập trung chính vào các khía cạnh là đánh giá bảo mật và hiệu năng thực thi truy vấn của các lược đồ cũng như tính khả thi triển khai lược đồ truy vấn.

- So sánh hiệu quả với các công trình nghiên cứu liên quan nổi bật trước đó.

3. PHƯƠNG PHÁP NGHIÊN CỨU

Hệ thống danh mục công trình nghiên cứu tiêu biểu liên quan đến đề tài. Từ đó phân tích chỉ ra các tồn tại nhằm xác định được đối tượng, phạm vi chi tiết, mục tiêu và nội dung nghiên cứu cụ thể của luận án.

Phát triển các phương pháp giải quyết bài toán dựa trên các đề xuất cấu trúc dữ liệu mới kết hợp với lựa chọn áp dụng các điểm mạnh từ những nghiên cứu trước đó: Xây dựng cơ sở dẫn luận và ý tưởng giải quyết bài toán, sau đó hiện thực theo thứ tự: lựa chọn mô hình triển khai và đề xuất thiết kế của lược đồ, xây dựng các cấu trúc dữ liệu, quy trình truy vấn và thuật toán liên quan.

Lựa chọn CSDL và các kịch bản, chỉ số đo lường phù hợp để thực nghiệm và phân tích đánh giá hiệu quả của lược đồ đề xuất. Cuối cùng là so sánh sự hiệu quả với các nghiên cứu liên quan trước đó.

4. CÁC ĐÓNG GÓP CỦA LUẬN ÁN

Luận án có hai đóng góp chính là đề xuất hai lược đồ SSE tương ứng hỗ trợ truy vấn hiệu quả chuỗi con trên dữ liệu ký tự và truy vấn khoảng trên dữ liệu số trong CSDLQH mã hoá. Áp dụng nhất quán mô hình DAS-PROXY trong triển khai hai lược đồ đã đề xuất, mang lại tính khả thi và tiềm năng cao trong ứng dụng thực tiễn. Tính hiệu quả thể hiện qua khả năng đảm bảo các tiêu chí về bảo mật, hiệu năng và tính khả thi triển khai của các lược đồ. Cụ thể:

1) Đề xuất xây dựng lược đồ DIQ-SSE dựa trên chỉ mục mù hỗ trợ truy vấn chuỗi con hiệu quả trên dữ liệu ký tự trong CSDLQH mã hoá khi người dùng yêu cầu chuỗi điều kiện "$LIKE\ '\%\ substring\ \%'$". Điểm nổi bật của đóng góp này là xây dựng quy trình truy vấn tuần tự trên hai chỉ mục mù $Index1, Index2$ và sử dụng một số cấu trúc dữ liệu mới trong xây dựng $Index1, Index2$.

2) Đề xuất xây dựng lược đồ ESIT-SSE dựa trên chỉ mục mù hỗ trợ truy vấn khoảng hiệu quả trên dữ liệu số trong CSDLQH mã hoá khi được người dùng yêu cầu chuỗi điều kiện "$BETWEEN\ l\ and\ h$". Điểm nổi bật của đóng góp này là đưa ra quy trình xây dựng chỉ mục mù qua hai bước, gồm: Bước 1. Xây dựng chỉ mục NewBucketIndex; Bước 2. Biến đổi NewBucketIndex về vector giấu tin IHV cho phép che giấu thông tin thứ tự các chỉ mục.

5. BỐ CỤC CỦA LUẬN ÁN

Chương 1. Tổng quan về mã hoá có thể tìm kiếm và vấn đề nghiên cứu

Trong chương 1, luận án tập trung giới thiệu về lược đồ mã hoá có thể tìm kiếm, phân loại các lược đồ, các kỹ thuật tổ chức mã hoá - phương thức tìm kiếm trong xây dựng lược đồ, mô hình DAS-PROXY và tình hình nghiên cứu liên quan tới đề tài luận án. Từ đó nêu các nhận xét, bình luận, chỉ ra các khoảng trống nghiên cứu và kết luận các vấn đề cần giải quyết trong luận án. Trong chương này cũng trình bày một số kiến thức cơ sở phục vụ cho các nội dung sẽ trình bày tại chương 2, 3.

Chương 2. Phát triển phương pháp truy vấn chuỗi con hiệu quả trên dữ liệu ký tự trong CSDLQH mã hoá

Trong chương 2, luận án tập trung đề xuất xây dựng lược đồ DIQ-SSE và triển khai trên mô hình DAS-PROXY cho phép truy vấn chuỗi con hiệu quả trong CSDLQH mã hoá thuê ngoài thông qua chuỗi điều kiện "$LIKE\ '\%\ substring\ \%'$". Trong đó điểm nhấn là xây dựng cặp chỉ mục mù đặc biệt, các thuật toán liên quan cùng quy trình truy vấn hiệu quả cho lược đồ. Chương 2 cũng dành một phần lớn nội dung để phân tích bảo mật, thực nghiệm đánh giá hiệu năng, độ phức tạp tìm kiếm để khẳng định tính hiệu quả của lược đồ DIQ-SSE.

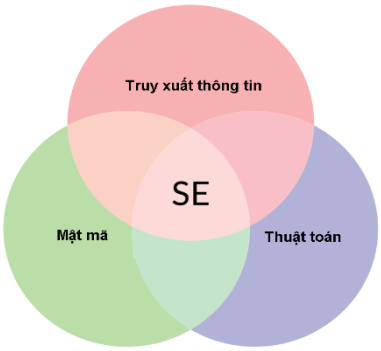
Chương 3. Phát triển phương pháp truy vấn khoảng hiệu quả trên dữ liệu số trong CSDLQH mã hoá

Trong chương 3, luận án tập trung đề xuất xây dựng lược đồ ESIT-SSE và triển khai trên mô hình DAS-PROXY cho phép truy vấn khoảng hiệu quả trong CSDLQH mã hoá thuê ngoài thông qua chuỗi điều kiện "$BETWEEN\ l\ and\ h$". Trọng tâm của quá trình này là kỹ thuật phân khoảng dữ liệu số và các phương pháp biến đổi cho phép tạo ra một chỉ mục đặc biệt (còn gọi là vector giấu tin IHV) cung cấp khả năng so sánh an toàn nhưng không tiết lộ nội dung và thứ tự chỉ mục. Tiếp theo đề xuất cấu trúc dữ liệu IHV_B$^+$Tree và thuật toán tìm kiếm cho phép tăng tốc truy vấn khoảng. Cuối chương tập trung làm rõ tính hiệu quả của lược đồ ESIT-SSE thông qua các thực nghiệm đánh giá hiệu năng, độ phức tạp tìm kiếm và phân tích bảo mật.

CHƯƠNG 1 . TỔNG QUAN VỀ MÃ HOÁ CÓ THỂ TÌM KIẾM VÀ VẤN ĐỀ NGHIÊN CỨU

1.1. TỔNG QUAN VỀ MÃ HOÁ CÓ THỂ TÌM KIẾM (SE)

Với các thách thức của bài toán mã hoá có thể tìm kiếm khi được triển khai tại các hạ tầng thuê ngoài, các nhà nghiên cứu đã cho ra đời một lĩnh vực mới là "Mã hóa có thể tìm kiếm (sau đây gọi là SE)", là sự kết hợp đặc biệt giữa các nghiên cứu về bảo mật, truy xuất thông tin và giải thuật tìm kiếm (xem **Hình 1.1**). **Bảng 1.1** chỉ thống kê một số hướng và công trình nghiên cứu tiêu biểu trong lĩnh vực SE theo dòng thời gian.



Hình 1.1. Các lĩnh vực liên quan trong SE

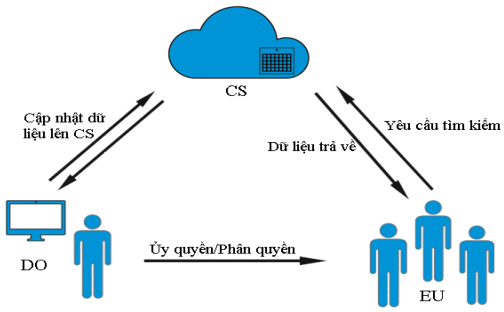
Bảng 1.1. Một số hướng nghiên cứu trong SE theo thời gian

Năm	**Hướng nghiên cứu tiêu biểu trong SE**
2000	Mã hóa đối xứng có thể tìm kiếm
2003	Chỉ mục mã hóa có thể tìm kiếm
2004	Mã hóa công khai có thể tìm kiếm
2008	Tìm kiếm đa từ khóa và tìm kiếm liên hợp các từ khóa
2010	Tìm kiếm từ khóa đơn có xếp hạng Tìm kiếm mờ
2011	Tìm kiếm đa từ khóa có xếp hạng
2013	Tìm kiếm dựa trên từ khóa đồng nghĩa Tìm kiếm dựa trên ngữ nghĩa của từ khóa
2014	Giải pháp SE dựa trên phần cứng
2015-2023	Chỉ mục tìm kiếm thích ứng; Tìm kiếm chuỗi con; Truy vấn dữ liệu mã dựa trên chia sẻ thông tin bí mật; Mã hóa đồng hình; …

1.2. TRIỂN KHAI LƯỢC ĐỒ SE TRÊN MÔ HÌNH DAS

1.2.1. Mô hình DAS

DAS là một mô hình cung cấp CSDL như một dịch vụ, mang tới khả năng quản lý dữ liệu từ xa đối với CSDL nói chung và cho phép triển khai hiệu quả các lược đồ SE trên CSDLQH mã nói riêng. Về cơ bản DAS gồm 3 đối tượng là DO, EU và CS. **Hình 1.2** mô tả mối quan hệ vận hành giữa ba đối tượng này.



Hình 1.2. Mô hình DAS

1.2.2. Áp dụng mô hình DAS vào triển khai lược đồ SE

Để đảm bảo nhiệm vụ của các đối tượng trong mô hình DAS, lược đồ SE cần được thiết kế từ 5 thành

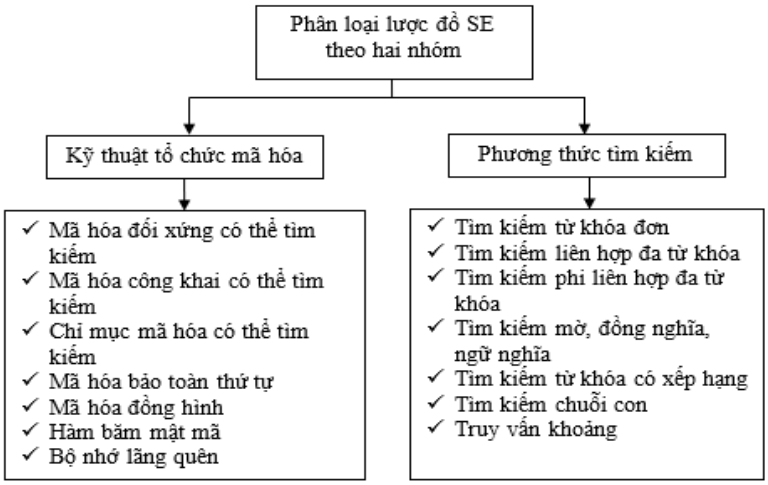
phần/thuật toán là: $KeyGen,\ BuildIndex,\ Trapdoor,\ Search,\ Decrypt$.

1.3. TRIỂN KHAI LƯỢC ĐỒ SE TRÊN MÔ HÌNH DAS-PROXY

Để khắc phục các tồn tại của DAS, ý tưởng sử dụng một máy chủ trung gian tin cậy (Proxy) để quản lý tất cả dữ liệu bí mật và thực thi các nghiệp vụ mà DO uỷ quyền đã được đề xuất trong nhiều nghiên cứu. Trong luận án này, máy chủ Proxy được vận dụng đưa vào trong mô hình DAS (gọi là mô hình DAS-PROXY) để triển khai thống nhất cho các lược đồ SE đề xuất tại chương 2 và 3. PROXY thể hiện vai trò lớp trung gian tin cậy giữa EU và CS. Mọi công việc cần đến tính riêng tư như: sinh khóa, quản lý khóa, tạo chỉ mục, biến đổi truy vấn, mã hóa, giải mã hay phân quyền, xác thực người dùng đều được tích hợp tại PROXY.

1.4. PHÂN LOẠI LƯỢC ĐỒ SE

Có nhiều cách tiếp cận khác nhau trong việc phân loại các lược đồ SE, trong đó **Hình 1.6** tổng hợp và trình bày phân loại các lược đồ SE theo góc nhìn về "kỹ thuật tổ chức mã hóa" và "phương thức tìm kiếm".



Hình 1.6. Phân loại lược đồ SE

1.5. MỘT SỐ KỸ THUẬT TỔ CHỨC MÃ HÓA TRONG THIẾT KẾ LƯỢC ĐỒ SE

1.5.1. Lược đồ mã hoá đối xứng có thể tìm kiếm (SSE)

Trong lược đồ mã hóa đối xứng có thể tìm kiếm (hay còn gọi là SSE), các tài liệu/bản ghi dữ liệu được mã hóa bằng cách sử dụng các khóa đối xứng riêng tư giữa DO và EU. Sử dụng khóa đối xứng kết hợp cùng các thuật toán đặc biệt, DO mã hóa tài liệu cùng với chỉ mục và gửi chúng đến CS, sau đó EU có thể truy cập dữ liệu được mã hóa bằng cách tạo cửa sập (Trapdoor) cho máy chủ đám mây. CS thực hiện tìm kiếm trên dữ liệu được mã hóa dựa trên thông tin từ cửa sập và gửi kết quả tới EU, sau đó, kết quả thu được có thể được giải mã bằng cách sử dụng khóa đối xứng.

1.5.2. Lược đồ mã hoá công khai có thể tìm kiếm (PKSE)

Các lược đồ mã hóa công khai có thể tìm kiếm (PKSE) sử dụng cặp khóa công khai và khóa riêng để mã hóa và giải mã. Trong lược đồ PKSE, DO mã hóa tài liệu cùng với chỉ mục tài liệu bằng khóa công khai của EU và lưu trữ lên CS. Khi đó EU có thể sử dụng khoá bí mật để tạo cửa sập (Trapdoor) và yêu cầu truy vấn trên CS, sau đó giải mã kết quả trả về bởi CS.

1.5.3. Lược đồ mã hoá có thể tìm kiếm dựa trên chỉ mục

Một trong các hướng tiếp cận hiệu quả với các lược đồ SSE và PKSE là giải pháp thay thế bản mã hỗ trợ tìm kiếm bằng chỉ mục mã hoá hỗ trợ tìm kiếm (còn gọi là chỉ mục mù). Chỉ mục mù có ưu điểm không thể giải mã, có tính bảo mật cao và cho phép kết hợp với các cấu trúc dữ liệu nhằm tăng tốc độ cũng như hỗ trợ đa dạng các phương thức truy vấn từ người dùng.

1.6. MỘT SỐ PHƯƠNG THỨC TÌM KIẾM TRONG CÁC LƯỢC ĐỒ SE

1.6.1. Tìm kiếm từ khóa đơn chính xác

Lược đồ SE hỗ trợ tìm kiếm từ khóa đơn cho phép EU chỉ đưa một từ khóa vào để truy vấn và trả về các tài liệu có chứa từ khóa này. Các lược đồ thường sử dụng danh sách các từ khóa hay được yêu cầu truy vấn để xây dựng thành các bảng chỉ mục hỗ trợ tìm kiếm (sử dụng cấu trúc chỉ mục chuyển tiếp hoặc đảo ngược).

1.6.2. Tìm kiếm đa từ khóa chính xác

- Tìm kiếm liên hợp đa từ khóa: cho phép EU đưa ra yêu cầu tìm kiếm nhiều từ khoá đồng thời, khi đó điều kiện truy vấn được biểu thị qua chuỗi liên hợp của các từ khóa $Q = \{kw_1\ and\ kw_2\ and\ \cdots\ and\ kw_n\}$ và kết quả trả về là những tài liệu/bản ghi có chứa các từ khóa trong chuỗi này.

- Tìm kiếm phi liên hợp đa từ khóa: cho phép truy xuất các tài liệu chứa tất cả hoặc một tập hợp con các từ khoá tìm kiếm.

1.6.3. Tìm kiếm từ khóa gần đúng

- Tìm kiếm mờ: cho phép tìm kiếm với các từ khóa có sai xót nhỏ về chính tả.

- Tìm kiếm dựa vào từ đồng nghĩa: cho phép tìm kiếm khi EU nhập từ khóa có nghĩa tương tự với từ khóa chuẩn trong từ điển.

- Tìm kiếm dựa vào ngữ nghĩa: cho phép tìm kiếm các bản ghi có chứa những từ khóa có mối quan hệ ngữ nghĩa với từ khóa mà EU yêu cầu.

1.6.4. Tìm kiếm chuỗi con

Tìm kiếm chuỗi con trên CSDLQH mã lại là một vấn đề phức tạp, khó thực hiện và ít nghiên cứu hơn so với tìm kiếm từ khoá. Có hai hướng tiếp cận để giải quyết bài toán này là: phân tích chuỗi con thành tập các từ khoá và tận dụng các lược đồ SE tìm kiếm theo từ khoá để tìm kiếm hoặc thiết kế các lược đồ SE với chức năng tìm kiếm chuỗi con chuyên biệt để thực hiện

1.6.5. Tìm kiếm khoảng

Tìm kiếm khoảng cũng là yêu cầu phổ biến của EU đối với truy xuất thông tin trong CSDLQH mã vì đây là dạng thức truy vấn tổng quát, bao hàm đầy đủ so sánh bằng, lớn hơn, nhỏ hơn trên dữ liệu số. Về cơ bản các lược đồ SE hỗ trợ truy vấn khoảng trên dữ liệu số mã hóa đều sử dụng chỉ mục mù.

1.7. CÁC YÊU CẦU ĐỐI VỚI LƯỢC ĐỒ MÃ HÓA CÓ THỂ TÌM KIẾM

Yêu cầu SE hướng tới là sự cân bằng của 4 yếu tố: "Bảo mật", "Hiệu năng", "Khả thi triển khai" và "Đa dạng khả năng truy vấn". Trong thực tế, hoàn toàn ta có thể tăng khả năng bảo mật lên cao tuy nhiên nó lại làm giảm hiệu năng cũng như khả năng đa dạng thực thi các lệnh truy vấn và ngược lại. Các tiêu chí đánh giá yếu tố bảo mật gồm: Tính riêng tư của tài liệu/bản ghi rõ; Tính riêng tư của chỉ mục; Tính riêng tư của mẫu truy vấn; Tính riêng tư của mẫu truy cập; Tính riêng tư của cửa sập. Các tiêu chí đánh giá yếu tố hiệu năng gồm: Chi phí mã hoá và xây dựng chỉ mục; Chi phí truy vấn; Chi phí truyền dữ liệu; Chi phí lưu trữ.

1.8. CÁC MÔ HÌNH BẢO MẬT CỦA LƯỢC ĐỒ SE

Cho tới nay, mô hình bảo mật thích ứng IND-CKA2 được đánh giá là mạnh mẽ nhất và là tiêu chuẩn cho các lược đồ SSE. Đây là một biến thể của IND-CKA1, trong đó kẻ gian có thể sử dụng thông tin của các truy vấn trước đó (tức là kết quả và cửa sập) để tạo ra các truy vấn mới để có được cái nhìn sâu sắc về các tài liệu và chỉ mục được mã hóa (mà không có giả định rằng tất cả các truy vấn được thực hiện cùng một lúc).

1.9. TÌNH HÌNH NGHIÊN CỨU LIÊN QUAN TỚI ĐỀ TÀI LUẬN ÁN

1.9.1. Tình hình nghiên cứu về truy vấn trên dữ liệu ký tự mã hóa

Phần lớn các nghiên cứu liên quan tới truy vấn trên dữ liệu ký tự mã hóa tập trung chính vào kỹ thuật tìm kiếm theo từ khóa. Tuy nhiên, thực tế người dùng lại có xu hướng gõ chuỗi con bất kỳ để tìm kiếm dữ liệu trên các hộp tìm kiếm, email hay tra cứu nội dung tài liệu, ... Tìm kiếm chuỗi con bao gồm một số dạng thức

phổ biến: tìm kiếm tiền tố, hậu tố và tìm kiếm chuỗi con độ dài bất kỳ trên dữ liệu mã. Các công trình tiêu biểu về tìm kiếm chuỗi con trên dữ liệu mã được trích dẫn và phân tích trong luận án đã chỉ ra một số tồn tại, khó khăn trong vấn đề chi phí lưu trữ cao (hầu hết cỡ $O(N)$), chưa có sự cân bằng giữa hiệu năng tìm kiếm (chi phí cỡ $O(l_{substr} + occ)$ đến $O({l_{substr}}^2 + occ)$ trên mỗi văn bản/bản ghi) và bảo mật thông tin (rò rỉ thông tin từ chỉ mục, mẫu truy vấn, ...). Các nghiên cứu phơi bày vấn đề đánh đổi giữa hiệu năng và bảo mật, tạo ra thách thức về tính khả thi trong triển khai thực tiễn. Vì vậy, một trong các nội dung quan trọng được tập trung nghiên cứu và trình bày trong luận án này là đề xuất phương pháp truy vấn chuỗi con hiệu quả trên dữ liệu ký tự mã hóa trong CSDLQH mã, tính hiệu quả được tập trung thể hiện thông qua các tiêu chí là: mô hình triển khai khả thi, đảm bảo cân bằng hiệu năng truy vấn và bảo mật thông tin của lược đồ tìm kiếm.

1.9.2. Tình hình nghiên cứu về truy vấn trên dữ liệu số mã hóa

Các giải pháp truy vấn khoảng thường tồn tại các vấn đề như: i) để đảm bảo nâng cao tính bảo mật thì lại đánh đổi bởi chi phí tính toán và tìm kiếm quá lớn hoặc ii) chấp nhận rò rỉ quá nhiều thông tin bản rõ để tăng hiệu năng tính toán hoặc iii) cho phép kết quả truy vấn dư thừa quá nhiều dữ liệu nhằm hạn chế lộ thông tin bản rõ. Cụ thể xét hai hướng tiếp cận chính trong xử lý truy vấn khoảng trên dữ liệu số để thấy các ưu nhược điểm như sau:

- Hướng thứ nhất tận dụng các lược đồ tìm kiếm theo từ khoá để phục vụ truy xuất thông tin rời rạc về các phần tử hoặc khoảng con nằm trong khoảng cần truy vấn. Tuy nhiên, việc phân rã khoảng truy vấn thành các đối tượng nhỏ hơn là phức tạp, không mang tính tổng quát và không đáp ứng tốt trong trường hợp xuất hiện các yêu cầu truy vấn khoảng mới chưa từng được thống kê. Hệ quả quan trọng nhất là gây tăng chi phí tìm kiếm do phải thực hiện nhiều lần khi tìm kiếm từ khoá tương ứng với các giá trị số/khoảng con mà nó đại diện. Để cải thiện hiệu năng, các nghiên cứu theo hướng này tích hợp các cấu trúc dữ liệu liên quan (cây nhị phân, bộ lọc Bloom, …) nhằm xây dựng lên các chỉ mục hỗ trợ tìm kiếm hiệu quả, tuy nhiên các cấu trúc này còn tồn tại các vấn đề về rò rỉ thông tin bản rõ.

- Hướng thứ hai sử dụng các kỹ thuật tổ chức mã hoá cho phép biến đổi trực tiếp các giá trị số về dạng chỉ mục hỗ trợ các phép so sánh. Ưu điểm chung của các kỹ thuật này là tạo ra các chỉ mục hỗ trợ triển khai truy vấn dữ liệu số thuận lợi, tuy nhiên tồn tại một số vấn đề như: bảo toàn thứ tự gây rò rỉ thông tin bản rõ (giải pháp OPE), kết quả trả về tồn tại bản ghi dư thừa (giải pháp Bucket), chi phí xây dựng chỉ mục và tìm kiếm trên chỉ mục cao (giải pháp FHE).

1.10. KẾT LUẬN CHƯƠNG 1

Chương 1 đã tổng quan bức tranh mã hoá có thể tìm kiếm (SE), mô hình DAS-PROXY, các kỹ thuật tổ chức mã hoá cùng những phương thức tìm kiếm phổ biến trong SE và tình hình nghiên cứu liên quan tới đề tài luận án. Đó là cơ sở để rút ra các nhận xét quan trọng sau đây nhằm định hướng nội dung nghiên cứu ở các chương tiếp theo, cụ thể:

- Nhận xét 1.1: Chọn lược đồ SSE xây dựng trên chỉ mục mù và triển khai lược đồ trên mô hình DAS-PROXY là giải pháp cho đề tài luận án.

- Nhận xét 1.2: Đề xuất phương pháp truy vấn chuỗi con hiệu quả trên dữ liệu ký tự mã hóa trong CSDLQH mã là mục tiêu thứ nhất cần giải quyết của luận án.

- Nhận xét 1.3: Đề xuất phương pháp truy vấn khoảng hiệu quả trên dữ liệu số mã hóa trong CSDLQH mã là mục tiêu thứ hai cần giải quyết của luận án.

- Tính hiệu quả được nhấn mạnh ở khả năng chống rò rỉ thông tin và hiệu năng thực thi truy vấn của các lược đồ đề xuất.

CHƯƠNG 2. GIẢI PHÁP TÍCH HỢP DỮ LIỆU TRONG DỰ ĐOÁN ĐÁP ỨNG ĐƠN THUỐC

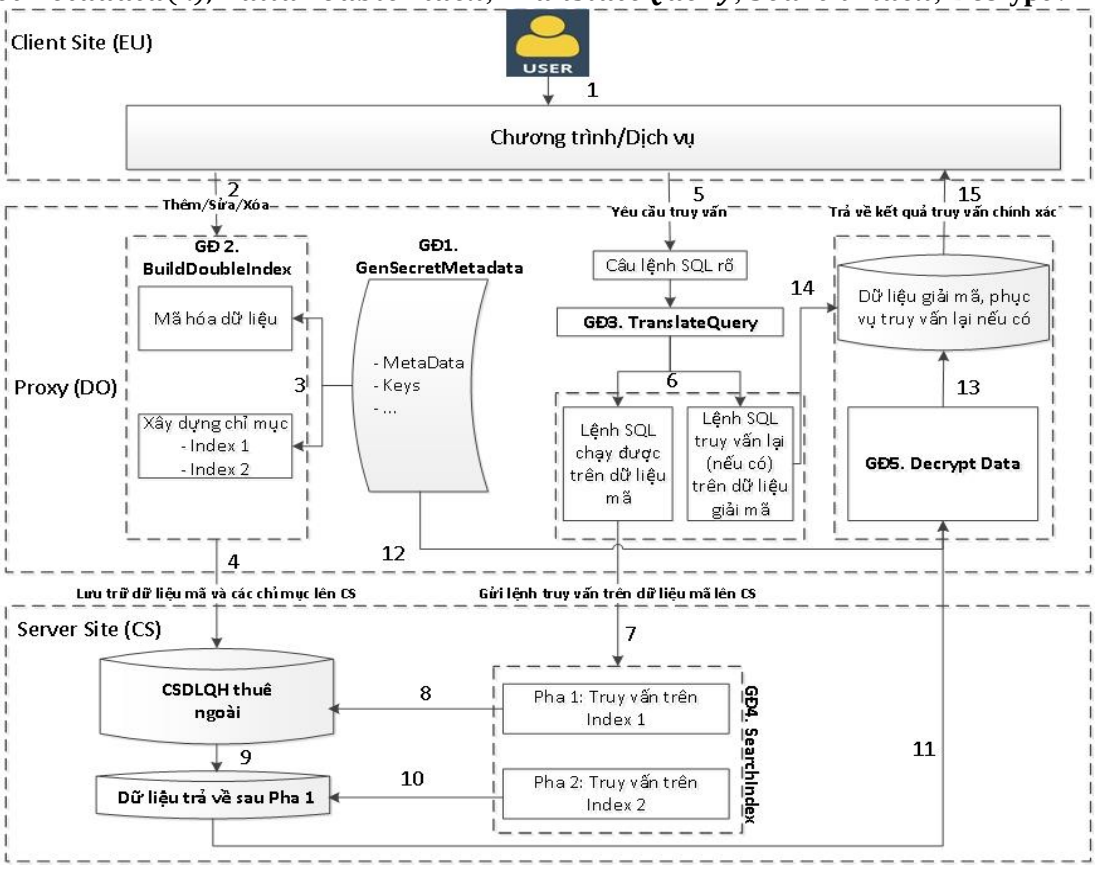
2.1. ĐẶT BÀI TOÁN VÀ DẪN LUẬN Ý TƯỞNG THỰC HIỆN

Quan hệ $R(ID, X_1, X_2, \ldots X_t, \ldots X_n)$ có số lượng bản ghi lớn với X_t là trường dữ liệu ký tự rõ nhạy cảm, quan hệ mã $R^E(ID, X_1, X_2, \ldots X_t^E, \ldots X_n)$ lưu trên đám mây tương ứng với quan hệ R với X_t^E là trường dữ liệu mã tương ứng của X_t. Cho một chuỗi ký tự con A_s' bất kỳ, yêu cầu: Truy vấn trên trường dữ liệu X_t^E của quan hệ R^E để tìm ra chính xác, hiệu quả những bản ghi có chứa A_s'.

Lựa chọn lược đồ SSE thiết kế dựa trên chỉ mục mù triển khai trên mô hình DAS-PROXY, giúp tăng cường tính khả thi triển khai, tăng tốc truy vấn và kế thừa tính bảo mật được đảm bảo theo mô hình IND-CKA2. Sử dụng phương pháp tìm kiếm từ khoá chính xác với hiệu năng cao để lọc bớt dữ liệu mã không thoả mãn, sau đó mới thực hiện truy vấn chuỗi con bằng các thuật toán đặc biệt trên phạm vi kết quả nhỏ hơn.

2.2. ĐỀ XUẤT LƯỢC ĐỒ VÀ MÔ HÌNH TRIỂN KHAI TRUY VẤN CHUỖI CON TRÊN CSDLQH MÃ

Đề xuất lược đồ DIQ-SSE hỗ trợ truy vấn chuỗi con hiệu quả (**Hình 2.1**) với 5 giai đoạn: $\boldsymbol{GenSecretMetadata(\lambda)}$; $\boldsymbol{BuildDoubleIndex}$; $\boldsymbol{TranslateQuery}$; $\boldsymbol{SearchIndex}$; **Decrypt** .



Hình 2.1. Triển khai lược đồ DIQ-SSE trên mô hình DAS-PROXY

2.3. XÂY DỰNG CẶP CHỈ MỤC HỖ TRỢ TÌM KIẾM

2.3.1. Xây dựng chỉ mục Index1

Định nghĩa 2.1. Cho $A = \{a_1, a_2, \ldots, a_z\}$ là tập các ký tự phân biệt để tạo nên các chuỗi rõ và $A_s = c_1 c_2 \ldots c_v$ là một chuỗi được tạo bởi các ký tự trong tập A ($z, v \geq 1$; $c_i \in A$ $với$ $1 \leq i \leq v$). Tập A_s^u gọi là tập các cặp ký tự kết nối theo khoảng cách u ($0 \leq u \leq v$) trên chuỗi A_s được định nghĩa như sau:

$A_s^u = A_s^{uFirstCharater} \cup A_s^{uSingleCharater} \cup A_s^{uPairCharater} \cup A_s^{uLastCharater}$ trong đó:

$$\begin{cases} A_s^{uFirstCharater} = \{c_1 || *\};\ A_s^{uSingleCharater} = \{c_i\};\ A_s^{uLastCharater} = \{* || c_v\};\ với\ 1 \leq i \leq v \\ A_s^{uPairCharater} = \{(c_i || j - i - 1 || c_j)\}\ với\ 1 \leq i < j \leq v\ và\ 0 \leq j - i - 1 \leq u \end{cases} \quad (2.1)$$

Định nghĩa 2.2. Cho A_s^u là tập các cặp ký tự kết nối theo khoảng cách u trên chuỗi ký tự A_s. Gọi $DA_s^u = \{DW_1, DW_2, \dots, DW_l\}$ là tập các cặp ký tự kết nối phân biệt theo khoảng cách u trên A_s nếu DA_s^u chỉ chứa l phần tử phân biệt của A_s^u.

Thuật toán 2.1: *BuildIndex1*

Input

$A_s = c_1c_2 \dots c_v$: chuỗi ký tự rõ nhạy cảm cần xây dựng chỉ mục $Index1$;
ID: giá trị định danh của bản ghi chứa chuỗi rõ A_s;
m: chiều dài chuỗi bit của $Index1$; k: số lượng hàm băm;
$H\_KEY = \{hKey_1, hKey_2, \dots, hKey_k\}$: tập khóa sử dụng cho k hàm băm;
u: khoảng cách trong các cặp ký tự kết nối; $nMax$: chiều dài của $f^{ElementMax}(D)$;

Output

$Index1$: là chuỗi bít BF với chiều dài m;

Các bước thực hiện

(1) Xây dựng tập $DA_s^u = \{DW_1, DW_2, \dots, DW_l\}$ theo **Định nghĩa 2.2**;

(2) Với mỗi từ khoá (một cặp ký tự kết nối) $DW_i \in DA_s^u \mid i \in [1, l]$, tính toán:

- Tập giá trị $\{p_1, p_2, \dots, p_k\}$ như sau: $\{p_1 = h_1(hKey_1, DW_i), p_2 = h_2(hKey_2, DW_i), \dots, p_k = h_k(hKey_k, DW_i)\}$;
- Tập giá trị $\{q_1, q_2, \dots, q_k\}$ như sau: $\{q_1 = h_1(p_1, ID), q_2 = h_2(p_2, ID), \dots, q_k = h_k(p_k, ID)\}$;
- Đặt $BF[q_j] = 1$ với $j \in [1, k]$;

(3) Sinh số ngẫu nhiên t trong khoảng $[0,\ nMax - l]$ thông qua hàm PRF, sau đó thực hiện:

- Tính $t' = t \bmod m$; $BF = BF \vee t'$;

(4) Trả về BF;

2.3.2. Xây dựng chỉ mục Index2

Định nghĩa 2.3. Cho $A = \{a_1,\ a_2, \dots, a_z\}$ là tập các ký tự phân biệt để tạo nên các chuỗi rõ và $A_s = c_1c_2 \dots c_v$ là một chuỗi được tạo bởi các ký tự trong tập A $(z, v \geq 1;\ c_i \in A\ với\ 1 \leq i \leq v)$. Tập $C = \{a_1, a_2 \dots, a_g\}$ là tập các ký tự phân biệt của A_s $(1 \leq g \leq z,\ g \leq v)$. Gọi $PA_s = \{Pa_1,\ Pa_2,\ \dots, Pa_g\}$ là tập các mảng nhị phân biểu diễn vị trí ký tự của chuỗi A_s nếu thỏa mãn biểu thức (2.2):

$$\begin{cases} 1 \leq i \leq v, 1 \leq j \leq g;\ Pa_j \text{ là mảng } v\ bits \text{ biểu diễn các vị trí của ký tự } a_j \text{ trong chuỗi } A_s; \\ Pa_j[i] = 1 \text{ khi } a_j = c_{v-i+1}; \end{cases} \quad (2.2)$$

Định nghĩa 2.4. Cho $A = \{a_1,\ a_2, \dots, a_z\}$ là tập các ký tự phân biệt sử dụng tạo nên các chuỗi rõ, cho q là một số nguyên tố lớn, tập $VA = \{Va_1,\ Va_2, \dots, Va_z\}$ được gọi là tập chứa các giá trị bí mật đại diện cho các ký tự của tập A dựa trên q nếu:

$$\begin{cases} \forall\ i, j\ (1 \leq i, j \leq z), Va_i \text{ và } Va_j \text{ là các số nguyên lớn hơn } q; \\ 1 \leq \overline{Va_i - Va_j} \leq q/2; \end{cases} \quad (2.4)$$

Định nghĩa 2.5. Cho $PA_s = \{Pa_1,\ Pa_2,\ \dots, Pa_g\}$ $(1 \leq g \leq v)$ là tập các mảng nhị phân biểu diễn vị trí ký tự của chuỗi A_s = "$c_1c_2 \dots c_v$". Cho $VA = \{Va_1,\ Va_2, \dots, Va_z\}$ là tập chứa các giá trị bí mật đại diện cho các ký tự của tập $A = \{a_1,\ a_2, \dots, a_z\}$ dựa trên số nguyên tố lớn q. Chỉ mục $Index2$ của chuỗi A_s là tập $IP_{A_s} = \{(I_{a_1}, IPa_1), (I_{a_2}, IPa_2), \dots, (I_{a_g}, IPa_g)\}$ với giá trị phần tử (I_{a_i}, IPa_i^N) được xác định như sau:

$$\begin{cases} 1 \leq i \leq g\ ;\ I_{a_i} = Va_i + k_i * q \text{ với } Va_i \in VA,\ k_i \text{ ngẫu nhiên riêng biệt cho mỗi } I_{a_i}; \\ IPa_i = Pa_i^{\gg k_{A_s} + Va_i} \text{ với } k_{A_s} \text{ bí mật dùng chung cho mọi } IPa_i; \end{cases} \quad (2.5)$$

Thuật toán 2.2. *BuildIndex2*

Input

A_s = "$c_1c_2 \dots c_v$": chuỗi ký tự rõ đã được gây nhiễu; q: số nguyên tố lớn;
$VA = \{Va_1,\ Va_2, \dots, Va_z\}$: tập các giá trị bí mật đại diện cho các ký tự phân biệt;

Output

Tập IP_{A_s}: chỉ mục $Index2$ của chuỗi A_s;

Các bước thực hiện

(1) Xác định $C = \{a_1, a_2 \dots, a_g\}$ là tập các ký tự phân biệt của $A_s, 1 \le g \le v$;

(2) Xây dựng $PA_s = \{Pa_1, Pa_2, \dots, Pa_g\}$;

(3) Xây dựng tập $IP_{A_s} = \{(I_{a_1}, IPa_1), (I_{a_2}, IPa_2), \dots, (I_{a_g}, IPa_g)\}$ *với* $1 \le i \le g$

$k_{A_s} \leftarrow PRF(\lambda)$ *với* λ *là tham số bảo mật*;

For $i = 1$ ***to*** g

Begin

$k_i \leftarrow PRF(\lambda)$ *với* λ *là tham số bảo mật*;

$Va_i \leftarrow f^v(a_i, VA)$ với f^v là hàm trả về phần tử Va_i trong VA mà đại diện cho ký tự a_i;

$I_{a_i} = Va_i + k_i * q$; $IPa_i^N = Pa_i^{N \gg k_{A_s} + Va_i}$;

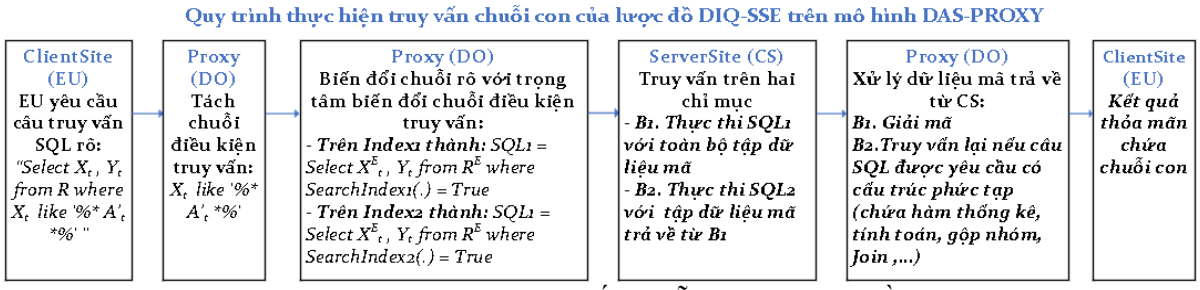
End

(4) Return IP_{A_s};

2.4. TRUY VẤN CHUỖI CON TRÊN CẶP CHỈ MỤC (INDEX1, INDEX2)

2.4.1. Quy trình thực hiện truy vấn chuỗi con

Theo mô hình đề xuất như **Hình 2.1**, quá trình truy vấn chuỗi con A'_s trên CSDLQH mã được thực hiện dựa trên 4 bước như **Hình 2.2**:



Hình 2.2. Quy trình thực hiện truy vấn chuỗi con trên lược đồ DIQ-SSE

2.4.2. Biến đổi điều kiện truy vấn cho phép tìm kiếm trên chỉ mục *Index*1

Thuật toán 2.3: *Tran*1

Input

$A'_s = c'_1 c'_2 \dots c'_t$: chuỗi con tìm kiếm; k: số lượng hàm băm;

$Type$: giá trị phân loại bằng 1 hoặc 2 hoặc 3 tương ứng với điều kiện truy vấn sau từ khóa $LIKE$ là "$\boldsymbol{A'_s}$ %" hoặc "%$\boldsymbol{A'_s}$ " hoặc "%$\boldsymbol{A'_s}$%";

$H\_KEY = \{hKey_1, hKey_2, \dots, hKey_k\}$: tập khóa sử dụng cho k hàm băm;

u: khoảng cách tối đa giữa các ký tự trong các cặp ký tự kết nối;

Output

Trả về tập $I1 = \{DW_1^{tran}, DW_2^{tran}, \dots, DW_{l'}^{tran}\}$ $(1 \le i \le l')$ với mỗi $DW_i^{tran} = \{p_1, p_2, \dots, p_{k'}\}$ $(1 \le j \le k',\ 1 \le k' \le k,\ p_j$ *là số nguyên*);

Các bước thực hiện

(1) Chọn ngẫu nhiên số u', k' sao cho $0 \le u' \le |A'_s|,\ u' \le u,\ 1 \le k' \le k$;

(2) Chọn ngẫu nhiên k' khóa từ tập H_KEY, ta được tập khóa $H\_KEY' = \{hKey'_1, hKey'_2, \dots, hKey'_{k'}\}$, $H_{KEY}{}' \subset$ H_KEY;

(3) Kiểm tra giá trị tham số $type$ để xây dựng $DA'_{s(.)}{}^{u'} = \{DW_1, DW_2, \dots, DW_{l'}\}$ là *tập các cặp ký tự kết nối phân biệt theo khoảng cách* u' *trên* A'_s cho phù hợp

If $Type = 1$ ***then*** $DA'_{s(.)}{}^{u'} = DA'_{s(left)}{}^{u'}$

Else If $Type = 2$ ***then*** $DA'_{s(.)}{}^{u'} = DA'_{s(right)}{}^{u'}$

Else $DA'_{s(.)}{}^{u'} = DA'_{s(mid)}{}^{u'}$

(4) Với mỗi chuỗi $DW_i \in DA'_{s(.)}{}^{u'}$ với $1 \le i \le l'$, tính toán:

$DW_i^{tran} = \{p_1, p_2, \dots, p_{k'}\}$ như sau: $\{p_1 = h_1(hKey'_1, DW_i),\ p_2 = h_2(hKey'_2, DW_i), \dots, p_{k'} = h_{k'}(hKey'_{k'}, DW_i)\}$

(5) Trả về tập $I1 = \{DW_1^{tran}, DW_2^{tran}, \dots, DW_{l'}^{tran}\}$

2.4.3. Biến đổi điều kiện truy vấn cho phép tìm kiếm trên chỉ mục $Index2$

Thuật toán 2.4: $Tran2$

Input

$A'_s = c'_1 c'_2 \dots c'_t$: chuỗi con tìm kiếm; p: số nguyên tố lớn;

$VA = \{Va_1, Va_2, \dots, Va_z\}$: tập giá trị bí mật đại diện cho các ký tự phân biệt của chuỗi rõ;

Output

Trả về tập $I2 = \{I_{c'_1}, I_{c'_2}, \dots, I_{c'_t}\}$ với $t = |A'_s|$;

Các bước thực hiện

(1) Set $t = |A'_s|$;

(2) Chuyển chuỗi A'_s thành tập các ký tự đơn $C = \{c'_1, c'_2, \dots, c'_t\}$

(3) Set $I2 = \left\{I_{c'_1}, I_{c'_2}, \dots, I_{c'_t}\right\}$ *với* $I_{c'_i} = 0$ $(1 \le i \le t)$, sau đó thực hiện:

For $i = 1$ ***to*** t

Begin

$k'_i \leftarrow PRF(\lambda)$ với λ là tham số bảo mật;

$Vc'_i \leftarrow f^v(c'_i, VA)$ với f^v là hàm trả về phần tử Va_j của mảng VA sao cho $a_j = c'_i$ $(1 \le j \le z)$; $I_{c'_i} = Vc'_i + k'_i * q$;

End

(4) Return $I2$;

2.4.4. Thực thi tìm kiếm trên chỉ mục $Index1$

Thuật toán 2.5: $SearchIndex1$

Input

$I1 = \{DW_1^{tran}, DW_2^{tran}, \dots, DW_{l'}^{tran}\}$: kết quả trả về của hàm $\boldsymbol{Trans1}$;

ID: giá trị định danh của bản ghi cần kiểm tra có chứa A'_s hay không;

BF: Giá trị $Index1$ tại bản ghi có định danh ID;

m: chiều dài chuỗi bit BF';

Output

$True$: bản ghi định có danh ID chứa chuỗi A'_s;

$False$: bản ghi định có danh ID không chứa chuỗi A'_s;

Các bước thực hiện

(1) Khởi tạo chuỗi BF' với m bits, đặt tất cả các bit của BF' bằng 0;

(2) Với mỗi $DW_i^{tran} \in I1$, $DW_i^{tran} = \{p_1, p_2, \dots, p_{k'}\}$ $(0 \le i \le l')$, thực hiện:

- Tính tập giá trị $\{q_1, q_2, \dots, q_{k'}\}$ sao cho: $q_1 = h_1(p_1, ID)$, $q_2 = h_2(p_2, ID)$, $\dots$, $q_{k'} = h_k(p_{k'}, ID)\}$; Đặt $BF'[q_j] = 1$ với $1 \le j \le k'$;

(3) Kiểm tra BF' có thuộc BF hay không:

If $BF' = BF' \wedge BF$ ***then*** $Result = True$

Else $Result = False$;

(4) Return $Result$;

2.4.5. Thực thi tìm kiếm trên chỉ mục $Index2$

Thuật toán 2.6: $SearchIndex2$

Input

$I2 = \{I_{c'_1}, I_{c'_2}, \dots, I_{c'_t}\}$: kết quả thực thi $\boldsymbol{Trans2}(.)$ với chuỗi con A'_s làm đầu vào;

$IP_{A_s} = \left\{\left(I_{a_1}, IPa_1\right), \left(I_{a_2}, IPa_2\right), \dots, \left(I_{a_g}, IPa_g\right)\right\}$: giá trị $Index2$ đại diện cho A_s;

Output

$True$: nếu A'_s là chuỗi con của A_s; $False$: nếu A'_s không là chuỗi con của A_s;

Các bước thực hiện

(1) Xây dựng tập $SB = \{(I_{c'_1}, IPc'_1), (I_{c'_2}, IPc'_2), \dots, (I_{c'_t}, IPc'_t)\}$ với mỗi phần tử $\left(I_{c'_i}, IPc'_i\right)$ sẽ tương ứng với một phần tử $\left(I_{a_j}, IPa_j\right) \in IP_{A_s}$ $(1 \le i \le t, 1 \le j \le v)$. Mã giả như sau:

$SB = \{\emptyset\}$;

$For\ i = 1\ to\ t$

$Begin$

$For\ j = 1\ to\ g$

$Begin$
$\quad$ $Begin$
$\quad\quad$ $\boldsymbol{if}\ \left(I_{c_i'} - I_{a_j}\right) mod\ q = 0\ \boldsymbol{then}$
$\quad\quad\quad$ $Begin$
$\quad\quad\quad\quad$ $I_{c_i'} = I_{a_j}; IPc_i' = IPa_j; SB = SB.Add((I_{c_i'}, IPc_i'));$
$\quad\quad\quad\quad$ $Break;$
$\quad\quad\quad$ End
$\quad$ End
End

(2) Biến đổi tập SB bằng cách giữ cố định IPc_1' và với mỗi IPc_i' thực quay vòng bít trái hoặc quay vòng bít phải một lượng $bias, (2 \le i \le t)$. Mã giả như sau:

$For\ i = 2\ to\ t$
$\quad Begin$
$\quad\quad \theta = (I_{c_1'} - I_{c_i'})\ mod\ q;$
$\quad\quad \boldsymbol{if}\ \frac{q}{2} \le \theta < q\ \boldsymbol{then}\ bias = -1 * \left(\theta - \frac{q}{2}\right);\ IPc_i'^N = IPc_i'^{N^{\ll \overline{bias}}};$
$\quad\quad \boldsymbol{Else\ if}\ 0 \le \theta < \frac{q}{2}\ \boldsymbol{then}\ bias = \theta;\ IPc_i'^N = IPc_i'^{N^{\gg bias}};$
$\quad End$

(3) Kiểm tra A_s' có là chuỗi con của A_s hay không:

$\quad \boldsymbol{If}\ IPc_1'^{\gg 1} \wedge IPc_2'^{\gg 2} \wedge \ldots \wedge IPc_i'^{\gg i} \wedge \ldots \wedge IPc_t'^{\gg t} \ne 0\ \boldsymbol{then}\ Result = True$
$\quad \boldsymbol{Else}\ Result = False$

(4) Return $Result$

2.5. Phân tích bảo mật của lược đồ DIQ-SSE

DIQ-SSE được coi đáp ứng mô hình bảo mật IND-CKA2 do kế thừa thiết kế từ một số lược đồ SSE tiêu chuẩn. Do đó, trong mục này luận án sẽ chỉ bổ sung, làm rõ thêm một số luận điểm về bảo mật của DIQ-SSE.

2.5.1. Phân tích bảo mật cặp chỉ mục

Xét trường hợp thuận lợi nhất cho kẻ gian, giả định họ biết trước: Thứ nhất: biết phương pháp xây dựng $Index1$, $Index2$ (nhưng không biết các $metadata$); Thứ hai: biết trước tập các chuỗi rõ P và tập mẫu PI_1 (chứa $Index1$), PI_2 (chứa $Index2$) tương ứng với một tập con P' của P (nhưng kẻ gian không biết P').

2.5.1.1. Phân tích bảo mật của $Index1$

Định nghĩa 2.6. Cho $A = \{a_1, a_2, \ldots, a_z\}$ là tập z ký tự phân biệt tạo nên các chuỗi rõ ($z \ge 1$). Tập D là tập các cặp ký tự phân biệt kết nối theo khoảng cách u ($0 \le u \le z-2$) trên tập ký tự A được định nghĩa như sau: $D = D^{FirtCharater} \cup D^{SingleCharater} \cup D^{PairCharater} \cup D^{LastCharater}$ trong đó:

$$\begin{cases} D^{FirtCharater} = \{a_i || *\};\ D^{SingleCharater} = \{a_i\};\ D^{LastCharater} = \{* || a_i\} \text{ với } 1 \le i \le z \\ D^{PairCharater} = \{(a_i || k || a_j)\} \text{ với } 1 \le i, j \le z;\ 0 \le k \le u \end{cases} \quad (2.9)$$

a) Kịch bản 1

Thống kê tần suất mỗi phần tử của D trên tập P và thống kê tần suất của các bit 1 tại mỗi vị trí trên các $Index1$, rồi sau đó so sánh chúng với nhau để phát hiện mỗi vị trí bit 1 của $Index1$ sẽ ứng với đặc tính nào trong A_s. Tuy nhiên thực tế các tham số $\boldsymbol{u}, \boldsymbol{u'}, \boldsymbol{k}$ đều lớn hơn $\mathbf{1}$ và bí mật, dẫn đến số lượng phần tử trong tập D không đoán được và số lượng bit đại diện cho một phần tử của tập DA_s^u sẽ nhiều hơn một bit.

b) Kịch bản 2

Dựa trên quan sát: chiều dài của chuỗi rõ sẽ tỷ lệ thuận với số lượng phần tử của tập DA_s^u, dẫn đến chiều dài chuỗi rõ có xu hướng tỷ lệ thuận với số lượng bit 1 trên chuỗi $Index1$.Tuy nhiên theo thuật toán $\boldsymbol{BuildIndex1}$, khả năng sử dụng kịch bản tấn công này rất thấp, bởi các chuỗi $Index1$ đều có chung độ dài m, mỗi chuỗi $Index1$ đều được bổ sung thêm một lượng bit ngẫu nhiên nên việc thống kê số lượng bit 1 trong mỗi $Index1$ sẽ không chính xác.

2.5.1.2. Phân tích bảo mật của Index2

a) Kịch bản 1

Thống kê số lần xuất hiện của mỗi ký tự phân biệt trong từng chuỗi ký tự rõ của tập P, sau đó thống kê tiếp số lần xuất hiện của các bit có giá trị 1 trong từng chuỗi IPa_i trong mỗi chỉ mục $Index2$ thuộc tập PI_2, quá đó tìm sự tương quan về ký tự bản rõ và giá trị bit 1. Tuy nhiên, trong thuật toán $BuildIndex2$ sử dụng chuỗi A_s đã được gây nhiễu làm đầu vào, vì vậy số bit 1 trên mỗi chuỗi IPa_i không phản ánh được số lượng thực của ký tự a_i trong chuỗi A_s nguyên bản.

b) Kịch bản 2

Dựa trên việc kẻ gian tính khoảng cách giữa các vị trí xuất hiện của cùng một ký tự trong chuỗi rõ, sau đó so sánh tìm sự tương đồng với khoảng cách các bit 1 trên từng chuỗi IPa_j trong mỗi $Index2$. Tuy nhiên, thuật toán $BuildIndex2$ có hai yếu tố khiến kịch bản này khó thực hiện là việc gây nhiễu chuỗi A_s trước khi thực thi $BuildIndex2$ và việc thực hiện dịch vòng phải chuỗi Pa_j đi một lượng bí mật $k_{A_s} + Va_j$.

2.5.2. Phân tích bảo mật mẫu truy vấn

Dựa vào nhu cầu tìm kiếm các chuỗi con trong thực tiễn, kẻ gian sẽ đoán được $I1, I2$ có thể liên quan tới chuỗi con nào. Qua đó suy diễn các bản ghi mã hoá được trả về sau truy vấn chứa chuỗi con nào. Thuật toán $Tran1$ sử dụng ngẫu nhiên k' khóa, tương tự thuật toán $Tran2$ sử dụng số ngẫu nhiên k_i' nên kết quả là với cùng một mẫu truy vấn A_s' mỗi lần được yêu cầu sẽ gửi tới CS một cặp $(I1, I2)$ khác nhau.

2.6. THỰC NGHIỆM VÀ ĐÁNH GIÁ HIỆU NĂNG CỦA LƯỢC ĐỒ DIQ-SSE

Hiệu năng truy vấn đánh giá qua: tỷ lệ lọc (FIR) và tỷ lệ lỗi (FAR) của tập dữ liệu mã sau mỗi lần truy vấn trên $Index1$ và $Index2$; thời gian thực thi từ thời điểm EU yêu cầu cho tới khi nhận được kết quả.

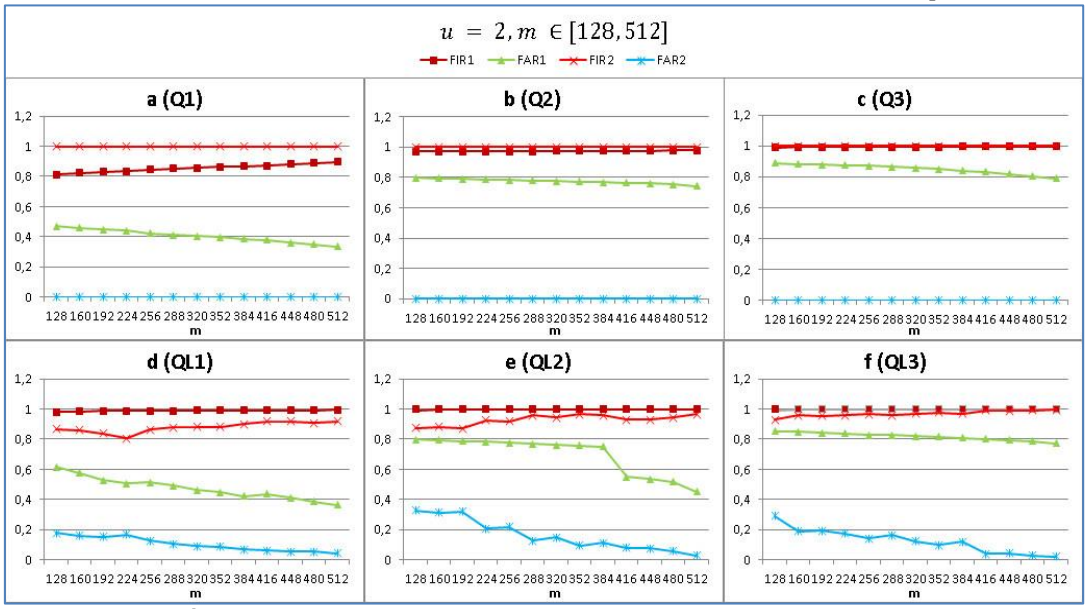
2.6.1. Chuẩn bị điều kiện thực nghiệm

Dữ liệu thử nghiệm là CSDLQH TPC-H. Luận án sử dụng bảng LINEITEM trong CSDL này làm đối tượng thực hiện các thử nghiệm. Xây dựng CSDLQH mã TPC_H_CS trong đó có bảng LINEITEM_CS có các cột L_COMMENT_E, L_COMMENT_SI1, L_COMMENT_SI2.

2.6.2. Thực nghiệm và đánh giá hiệu năng tìm kiếm

2.6.2.1. Kịch bản thực nghiệm 1

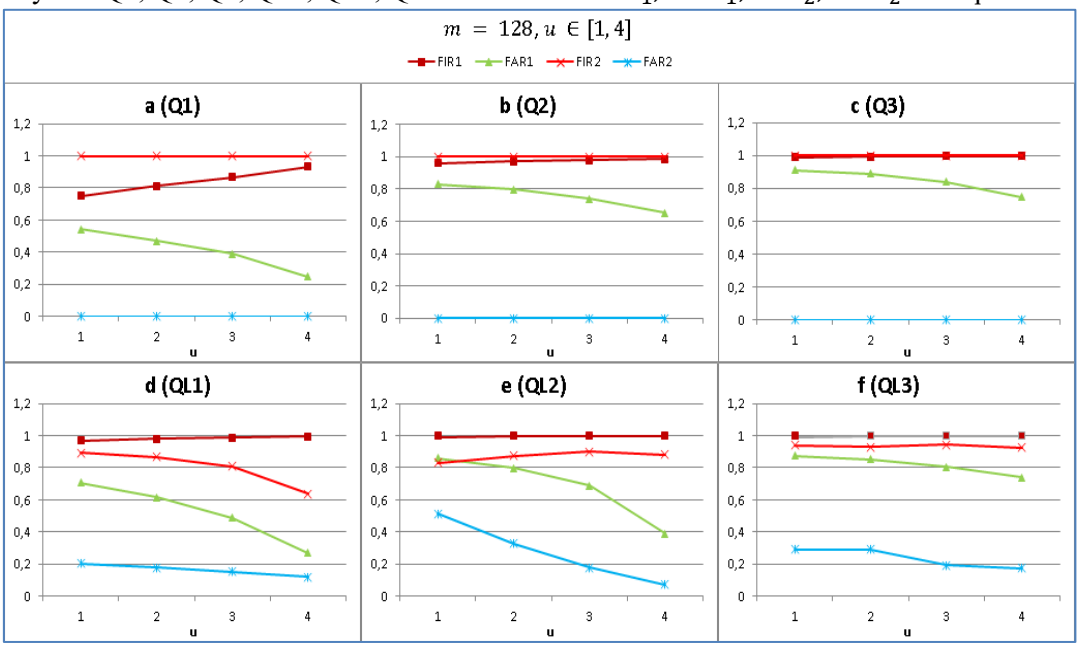
Kịch bản 1: Chọn $N_R = 1000000$, cố định $u = 2$, với m tăng dần từ 128 đến 512 lần lượt thực hiện các truy vấn Q1, Q2, Q3, QL1, QL2, QL3 và tính toán FIR_1, FAR_1, FIR_2, FAR_2. Kết quả như **Hình 2.5.**



Hình 2.5. Kết quả thực nghiệm kịch bản $N_R = 1000000, u = 2, m \in [128, 512]$

2.6.2.2. Kịch bản thực nghiệm 2

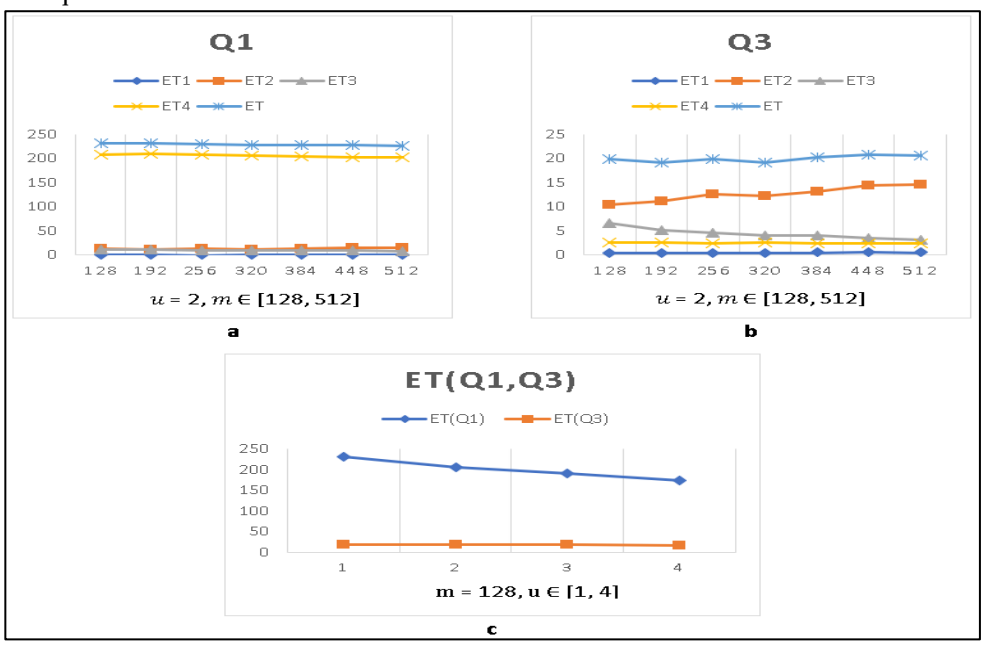
Kịch bản 2: Chọn $N_R = 1000000$, cố định $m = 128$, với u thay đổi ($u \in \{1,2,3,4\}$) lần lượt thực hiện các truy vấn Q1, Q2, Q3, QL1, QL2, QL3 và tính toán FIR_1, FAR_1, FIR_2, FAR_2. Kết quả như **Hình 2.6.**



Hình 2.6. Kết quả thực nghiệm kịch bản $N_R = 1000000$, $m = 128$, $u \in [1,4]$

2.6.3. Thực nghiệm và đánh giá thời gian truy vấn

Trong kịch bản thực nghiệm và đánh giá thời gian truy vấn của lược đồ, luận án sử dụng $u \in [1,4]$, $m \in [128,512]$ và lựa chọn các câu lệnh Q1, Q3 để đo thời gian thực thi cho 4 bước. Các giá trị thời gian ET_i được biểu diễn qua biểu đồ như **Hình 2.7.**



Hình 2.7. Đánh giá thời gian truy vấn Q1, Q3 với $u \in [1,4]$, $m \in [128,512]$

2.7. SO SÁNH HIỆU QUẢ LƯỢC ĐỒ DIQ-SSE VỚI MỘT SỐ NGHIÊN CỨU

Đánh giá chung: Các công trình liệt kê đều tiêu biểu, uy tín và liên quan tới lược đồ SE hỗ trợ truy vấn chuỗi con, trong đó nhiều tài liệu cập nhật được tình hình trong 5 năm trở lại đây. Cấu trúc dữ liệu hỗ trợ xây dựng chỉ mục trong lược đồ DIQ-SSE có tính mới so với các công trình khác khi đề xuất sử dụng tập DA_s^u và PA_s trong thiết kế cặp chỉ mục $Index1, Index2$.

Đánh giá về lược đồ DIQ-SSE khi so với các công trình liên quan:

- Các chi phí giao tiếp cũng như tính toán tại Proxy hay Client của DIQ-SSE là không đáng kể. Chi phí tìm kiếm chủ yếu tập trung tại CS qua việc thực thi hai thuật toán $SearchIndex1$ và $SearchIndex2$. Trong đó, thuật toán $SearchIndex1$ cần tới k lần tính toán trước khi thực hiện phép AndBitwise nên được đánh giá độ phức tạp cỡ $O(k)$. Thuật toán $SearchIndex2$ cần tới hai lần duyệt các ký tự chuỗi con nên độ phức tạp xấp xỉ cỡ $O(2.l_{substr})$. Thuật toán $SearchIndex1$ có chi phí rất tốt được áp dụng trên N bản ghi, trong khi đó thuật toán $SearchIndex2$ có độ phức tạp tính toán trung bình (không nổi trội hơn so với các công trình khác) nhưng lại có lợi thế do chỉ thực hiện trên tập dữ liệu nhỏ được trả về bởi $SearchIndex1$. Đây chính là yếu tố quan trọng giúp độ phức tạp tính toán tổng thể của lược đồ DIQ-SSE có ưu thế.

- Xét về số vòng giao tiếp thì DIQ-SSE cũng cho thấy ưu thế khi chỉ mất 1 vòng. Về khả năng chống rò rỉ chỉ mục và mẫu truy vấn cũng đạt yêu cầu. Qua đó cho thấy tính hiệu quả của lược đồ trên phương diện hiệu năng và bảo mật.

2.8. KẾT LUẬN CHƯƠNG 2

Chương 2 đề xuất lược đồ DIQ-SSE triển khai trên mô hình DAS-PROXY mang đến khả năng truy vấn chuỗi con hiệu quả trên CSDLQH mã. Cụ thể một số điểm nổi bật trong chương này:

- Thiết kế chỉ mục $Index1$ có ưu điểm chống rò rỉ thông tin và hỗ trợ truy vấn nhanh.
- Thiết kế chỉ mục $Index2$ với ưu điểm chống rò rỉ thông tin và tìm kiếm chính xác chuỗi con.
- Thiết kế quy trình truy vấn tuần tự qua hai chỉ mục $Index1$ và $Index2$.
- Thực nghiệm các kịch bản, so sánh với các công trình liên quan để làm rõ tính hiệu quả của DIQ-SSE.

Ngoài ra, một số nhược điểm cũng cần phải được ghi nhận gồm:

- Thiết kế của $Index1$ dẫn đến tồn tại bản ghi dương tính giả trong kết quả trả về sau truy vấn trên chỉ mục này. Trong khi đó $Index2$ mặc dù đã được làm nhiễu nhưng vẫn có thể để lộ ra một phần tính chất tuyến tính giữa độ dài chỉ mục với bản rõ khi xét trên các trường hợp chiều dài bản rõ quá dài hoặc quá ngắn.
- Tốc độ truy vấn trên $Index2$ bị ảnh hưởng bởi chiều dài chuỗi con.
- Chi phí lưu trữ chỉ mục của lược đồ DIQ-SSE chưa tốt, đạt mức $O(2N)$.

CHƯƠNG 3. PHÁT TRIỂN PHƯƠNG PHÁP TRUY VẤN KHOẢNG HIỆU QUẢ TRÊN DỮ LIỆU SỐ TRONG CSDLQH MÃ

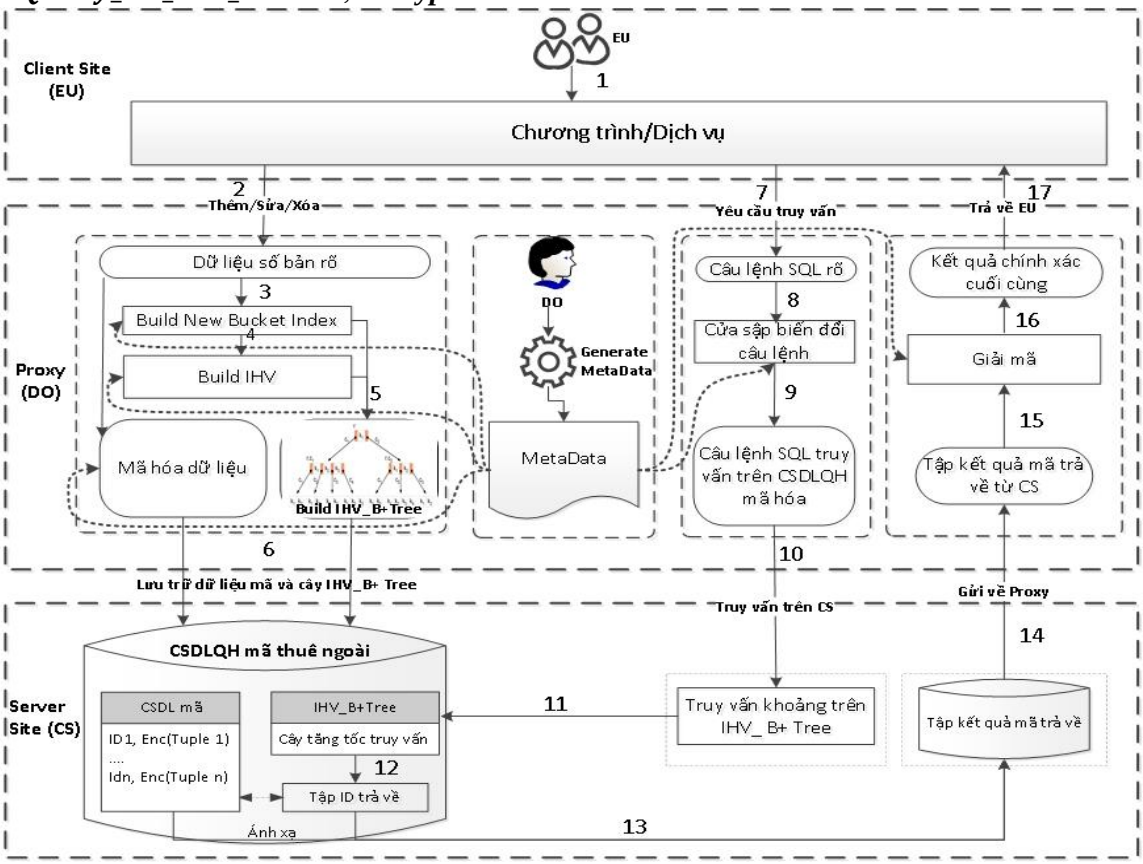
3.1. ĐẶT BÀI TOÁN VÀ DẪN LUẬN Ý TƯỞNG THỰC HIỆN

Quan hệ $R(ID, X_1, X_2,\ldots X_t, \ldots X_n)$ có số lượng bản ghi lớn với X_t là trường dữ liệu rõ dạng số nhạy cảm, quan hệ mã $R^E(ID, X_1, X_2,\ldots X_t^E, \ldots X_n)$ lưu trên đám mây tương ứng với quan hệ R với X_t^E là trường dữ liệu mã tương ứng của X_t. Cho một cặp giá trị số bất kỳ $(\boldsymbol{l}, \boldsymbol{h})$, yêu cầu: Truy vấn trên trường dữ liệu X_t^E của quan hệ R^E để tìm ra chính xác, hiệu quả những bản ghi thỏa mãn chuỗi điều kiện "X_t $BETWEEN$ l and h".

Luận án lựa chọn lược đồ SSE thiết kế dựa vào chỉ mục mù và triển khai trên mô hình DAS-PROXY nhằm tận dụng các thế mạnh của lược đồ SSE và chỉ mục tìm kiếm cũng như đảm bảo cơ sở về tính bảo mật theo mô hình IND-CKA2. Luận án đề xuất cải tiến phương pháp chia Bucket (tạo ra NewBucketIndex [CT5]), xây dựng phương pháp che giấu đặc tính thứ tự của các NewBucketIndex (gọi là IHV [CT4]) và thiết kế cấu trúc dữ liệu (gọi là cây IHV_B+ Tree) để tăng tốc truy vấn khoảng và loại bỏ dữ liệu dư thừa trả về.

3.2. ĐỀ XUẤT LƯỢC ĐỒ VÀ MÔ HÌNH TRIỂN KHAI TRUY VẤN KHOẢNG TRÊN CSDLQH MÃ

Đề xuất lược đồ ESIT-SSE hỗ trợ truy vấn khoảng hiệu quả gồm 5 giai đoạn (xem **Hình 3.1)**: $\boldsymbol{GenSecretMetadata(\lambda)}$; $\boldsymbol{Build\_IHV\_B^+Tree}$; $\boldsymbol{Transform\_RangeQueryCondition}$; $\boldsymbol{RangeQuery\_On\_IHV\_B^+Tree}$; ***Decrypt***.



Hình 3.1. Triển khai lược đồ ESIT-SSE trên mô hình DAS-PROXY

3.3. XÂY DỰNG CHỈ MỤC NEWBUCKETINDEX

3.3.1. Yêu cầu đối với chỉ mục NewBucketIndex

Các yêu cầu với chỉ mục NewBucketIndex: Một giá trị rõ có thể được đại diện bởi nhiều NewBucketIndex và ngược lại; Hỗ trợ chống các dạng tấn công: rò rỉ thông tin động và rò rỉ thông tin tĩnh; Giảm dư thừa dữ liệu trả về sau truy vấn; Thuận lợi biến đổi lệnh truy vấn; Hỗ trợ truy vấn hiệu suất cao.

3.3.2. Xây dựng chỉ mục NewBucketIndex

Định nghĩa 3.1. Cho $V = \{v_1,\ v_2, \dots, v_z\}$ là tập z các giá trị số lưu trong trường X_t của quan hệ R. Khi đó $D = \{d_1, d_2, \dots, d_n\}$ $(n \le z)$ được gọi là tập các phần tử phân biệt của tập V nếu các phần tử của D được xác định như sau: với $v_i, v_{i+1} \in V;\ v_i \le v_{i+1}$ và f_i là tần suất xuất hiện của v_i trong tập V thì:

$$d_i = \binom{f_i}{v_i},\ d_{i+1} = \binom{f_{i+1}}{v_{i+1}} \text{ với } 1 \le i \le n-1 \qquad (3.1)$$

Định nghĩa 3.2. Cho $D = \{d_1, d_2, \dots, d_n\}$ là tập các phần tử phân biệt của tập V và $d_i,\ d_{i+1} \in D$. Khi đó $P_i = \{p_{i1}, p_{i2}, \dots, p_{ik_i}\}$ $(1 \le i \le n)$ được gọi là tập giá trị ngẫu nhiên đại diện cho phần tử $d_i = \binom{f_i}{v_i}$ và tương tự P_{i+1} là tập giá trị ngẫu nhiên đại diện cho phần tử $d_{i+1} = \binom{f_{i+1}}{v_{i+1}}$. Tập $P_i,\ P_{i+1}$ phải có đặc điểm và mối quan hệ với nhau như sau:

- $[P_i^{Min},\ P_i^{Max}]$ là miền giá trị của tập P_i với $P_i^{Min},\ P_i^{Max}$ là các số nguyên và $P_i^{Max} - P_i^{Min} \ge f_i$;
- $p_{ij} \leftarrow PRF\left(P_i^{Min}, P_i^{Max}\right)$ với $1 \le j \le k_i,\ k_i \ge f_i$ và $P_i^{Min} \le p_{i1} \le p_{i2} \le \cdots \le p_{ik_i} \le P_i^{Max}$;
- $P_i^{Max} \le P_{i+1}^{Min}$ với $1 \le i \le n$;
- Số phần tử k_i của tập P_i được xác định theo nguyên tắc: $\frac{k_1}{f_1} \approx \frac{k_2}{f_2} \approx \cdots \approx \frac{k_i}{f_i} \approx \cdots \approx \frac{k_n}{f_n} \approx \varepsilon$ (3.2)

  Với ε gọi là hệ số mở rộng, ε nguyên và $\varepsilon \ge 1$.

Định nghĩa 3.3. Cho $V = \{v_1,\ v_2, \dots, v_z\}$ là tập z các giá trị số lưu trong trường X_t của quan hệ R, $D = \{d_1, d_2, \dots, d_n\}$ là tập các phần tử phân biệt của tập V và P_i $(1 \le i \le n)$ là tập giá trị ngẫu nhiên đại diện cho phần tử $d_i \in D$. Khi đó P được gọi là tập giá trị phẳng đại diện cho tập V nếu $P = P_1 \cup P_2 \cup \dots P_i \dots \cup P_n$.

Định nghĩa 3.4. Cho $V = \{v_1,\ v_2, \dots, v_z\}$ là tập z các giá trị số lưu trong trường X_t của quan hệ R và P được gọi là tập giá trị phẳng đại diện cho V. Tập $B = \{B_1, B_2, \dots., B_m\}$ được gọi là tập các giá trị NewBucketIndex đại diện cho tập V nếu:

- Tập P được chia ngẫu nhiên thành m khoảng liên tiếp nhau $(m \le |P|)$;
- Khoảng thứ j được đại diện bằng một giá trị nguyên ngẫu nhiên là B_j;
- Với $j, j' \in [1,\ m]$ và $j < j'$ thì $B_j \le B_{j'}$.

Thuật toán ***BuildNewBucketIndex*** xây dựng tập B và V^B như sau:

Thuật toán 3.1. *BuildNewBucketIndex*

Input

$V = \{v_1, v_2, \dots, v_z\}$: là tập z các giá trị số lưu trong trường X_t của quan hệ R;
ε: là hệ số mở rộng $(\varepsilon \ge 1)$; m: Số Bucket cần chia $(m \le |P|)$;

Output

Tập $V^B = \{v_1^B, v_2^B, \dots, v_z^B\}$: là tập các giá trị NewBucketIndex tương ứng với z phần tử trong V;

Các bước thực hiện

(1) Xây dựng tập $D = \{d_1, d_2, \dots d_i, \dots d_n\}$ là tập các phần tử phân biệt của tập V $(1 \le i \le n)$;

(2) Dựa trên hệ số ε, xây dựng $P_1, P_2, \dots, P_i, \dots P_n$ lần lượt là các tập giá trị ngẫu nhiên đại diện cho các phần tử $d_1, d_2, \dots d_i, \dots d_n$ $(1 \le i \le n)$;

(3) Xây dựng tập giá trị phẳng P đại diện cho tập V: $P = P_1 \cup P_2 \cup \dots \cup P_i \cup \dots \cup P_n$; Khi đó $P = \{p_1, p_2, \dots, p_{|P|}\}$;

(4) Chọn các giá trị biên mở rộng cho tập P là EP^{Min} và EP^{Max} sao cho $EP^{Min} \le p_1, EP^{Max} \ge p_{|P|}$;

(5) Xác định giá trị tại $m-1$ điểm giao nhau của m khoảng trên tập P bằng cách: Tính $m-1$ giá trị ngẫu nhiên phân biệt $\{ipv_1,\ ipv_2, \dots,\ ipv_{m-1}\}$ theo công thức: $ipv_t \leftarrow PRF\left(EP^{Min}, EP^{Max}\right)$ với $1 \le t \le m-1$ và $ipv_t < ipv_{t+1}$;

(6) Xác định tập $B = \{B_1, B_2, \dots B_t, \dots B_m\}$ là tập các giá trị NewBucketIndex đại diện cho V như sau:

$B_1 \leftarrow PRF\left(EP^{Min}, ipv_1\right)$, B_1 đại diện cho tập EP_1 với miền giá trị $[EP^{Min}, ipv_1)$;
$B_2 \leftarrow PRF(ipv_1, ipv_2)$, B_2 đại diện cho tập EP_2 với miền giá trị $[ipv_1, ipv_2)$;
…
$B_t \leftarrow PRF(\ ipv_{t-1}, ipv_t)$, B_t đại diện cho tập EP_t với miền giá trị $[ipv_{t-1}, ipv_t)$;

…

$B_m \leftarrow PRF(\ ipv_{m-1}, EP^{Max})$, B_m đại diện cho tập EP_m với miền giá trị $[ipv_{m-1}, EP^{Max})$;

(7) Dựa vào tập B, xác định tập $V^B = \{v_1^B, v_2^B, \dots, v_z^B\}$ tương ứng với V như sau:

(7.1) Với mỗi $v_j \in V$, xác định phần tử $d_i \in D$ sao cho $d_i = v_j$ $(1 \le j \le z;\ 1 \le i \le n)$;

(7.2) Xác định tập P_i tương ứng với d_i, lấy ngẫu nhiên giá trị $p \in P_i$;

(7.3) Kiểm tra nếu $p \in EP_t$ $(1 \le t \le m-1)$ thì B_t đại diện cho v_j, gán $v_j^B = B_t$;

(7.4) $V^B = \{v_1^B, v_2^B, \dots, v_z^B\}$ với $v_j^B \in B$ $(1 \le j \le z)$;

(8) Return V^B;

3.4. BIẾN ĐỔI GIÁ TRỊ NEWBUCKETINDEX THÀNH VECTOR GIẤU TIN (IHV)

3.4.1. Quy trình xây dựng vector giấu tin (IHV)

Xét điều kiện truy vấn trên dữ liệu rõ là "$\boldsymbol{X_t}$ *Between* $\boldsymbol{l}$ *And* $\boldsymbol{h}$. Khi đó nếu chỉ có NewBucketIndex, tại CS truy vấn được như mô tả như sau: "tìm kiếm các phần tử $v_i^B \in V^B$ sao cho $c_l \le v_i^B \le c_h$", với $c_l = Min(l^B)$ và $c_h = Max(h^B)$. Bài toán đặt ra là tại CS phải hỗ trợ cơ chế bảo mật để thực hiện được biểu thức so sánh $Min(l^B) \le v_i^B \le Max(h^B)$ nhưng không có cơ sở nào để thực hiện so sánh hai phần tử $v_i^B,\ v_j^B$ bất kỳ trong tập V^B. Như vậy, tại Proxy phải có giải pháp biến đổi tiếp v_i^B, $Min(l^B)$ và $Max(h^B)$ thành các dạng thức mới $\overrightarrow{ihv_b(v_i^B)}$, $\overrightarrow{ihv_c(Min(l^B))}$ và $\overrightarrow{ihv_c(Max(h^B))}$. Tổng quát ta xét biến đổi biểu thức $c_l \le v_i^B$.

Đặt $c = c_l$ và $b = v_i^B$, khi đó biểu thức $c_h \le v_i^B$ được thay thế thành $c \le b \iff (b, -1) \cdot (1, c) \ge 0$. Ký hiệu $\overrightarrow{O_b} = (b, -1)$ và $\overrightarrow{O_c} = (1, c)$ khi đó: $\overrightarrow{O_b} \cdot \overrightarrow{O_c} \ge 0$. Bước đầu che giấu thông tin thứ tự cho b và c trên CS dựa trên việc biến đổi $\overrightarrow{O_b}$ và $\overrightarrow{O_c}$ về các vector mới dài hơn nhưng vẫn đảm bảo tích vô hướng giữa chúng lớn hơn hoặc bằng 0. Ta thực hiện hoà nhập $\overrightarrow{O_b}$ và $\overrightarrow{O_c}$ lần lượt vào các vector trực giao $\overrightarrow{U_b} \perp \overrightarrow{U_c}$ tại các vị trí $pos1, pos2$ để tạo ra cặp vector nhúng $\overrightarrow{UO_b}$ và $\overrightarrow{UO_c}$. Để giải quyết vấn đề lộ vị trí $(pos1, pos2)$, luận án tiếp tục tăng cường bảo mật cho $\overrightarrow{UO_b}$, $\overrightarrow{UO_c}$ dựa trên ý tưởng chuyển đổi các vetor này về các ma trận cỡ kx1 sau đó nhân chúng lần lượt với các ma trận bí mật M_b, M_c cỡ kxk. Trong đó $M_c = M^T, M_b = M^{-1}$ (M là một ma trận bí mật khả nghịch, M_c là ma trận chuyển vị của M và M_b là ma trận nghịch đảo của M)

3.4.2. Thuật toán biến đổi NewBucketIndex thành vector giấu tin IHV

Thuật toán 3.2: ***Transform_NewBucketIndex_To_IHV***

Input

$b = v_i^B$: là giá trị NewBucketIndex đại diện cho giá trị rõ v_i; Vector $\overrightarrow{U_b}$ bí mật có chiều dài $k-2$;

Cặp vị trí bí mật $(pos1, pos2) |\ 1 \le pos1, pos2 \le k$; Ma trận bí mật M khả nghịch và có kích cỡ kxk;

Output

$\overrightarrow{ihv_b(b)}$ là vector giấu tin của giá trị b;

Các bước thực hiện:

Tạo vector $\overrightarrow{O_b} = (b, -1)$, sau đó thực hiện tiếp các bước sau:

(1) Hoà nhập vector $\overrightarrow{O_b}$ vào vector $\overrightarrow{U_b}$ tại các vị trí $(pos1, pos2)$ để nhận được vector nhúng $\overrightarrow{UO_b}$;

(2) Chọn ngẫu nhiên số nguyên dương r_b, tính lại $\overrightarrow{UO_b} = r_b * \overrightarrow{UO_b}$;

(3) Tính $M_b = M^{-1}$, biến đổi vector $\overrightarrow{UO_b}$ thành ma trận UO_b cỡ kx1;

(4) Xác định ma trận UO_b^M bằng công thức: $UO_b^M = M_b$x UO_b;

(5) Biến đổi ma trận UO_b^M về vector $\overrightarrow{UO_b^M}$;

(6) $\overrightarrow{ihv_b(b)} = \overrightarrow{UO_b^M}$;

(7) Return $\overrightarrow{ihv_b(b)}$;

3.4.3. Thuật toán biến đổi toán hạng so sánh thành vector giấu tin IHV

Thuật toán 3.3: ***Transform_ComparisonOperand_To_IHV***

Input

c: là các giá trị $c_l = Min(l^B)$ hoặc $c_h = Max(h^B)$; Vector cơ sở $\overrightarrow{U_c}$ bí mật có chiều dài $k-2$;

Cặp vị trí bí mật $(pos1, pos2) |\ 1 \le pos1, pos2 \le k$; Ma trận bí mật M khả nghịch và có kích cỡ kxk;

Output

$\overrightarrow{ihv_c(c)} = \overrightarrow{UO_c^M}$ là vector giấu tin của giá trị c;

Các bước thực hiện:

Tạo vector $\overrightarrow{O_c} = (1, c)$, sau đó thực hiện các bước sau:

(1) Hoà nhập vector $\overrightarrow{O_c}$ vào vector $\overrightarrow{U_c}$ tại các vị trí $(pos1, pos2)$ để nhận được vector nhúng $\overrightarrow{UO_c}$;

(2) Chọn ngẫu nhiên số nguyên dương r_c, tính lại $\overrightarrow{UO_c} = r_c * \overrightarrow{UO_c}$;

(3) Tính $M_c = M^T$, biến đổi vector $\overrightarrow{UO_c}$ thành ma trận UO_c cỡ kx1;

(4) Xác định ma trận UO_c^M bằng công thức: $UO_c^M = M_c$x UO_c;

(5) Biến đổi ma trận UO_c^M về vector $\overrightarrow{UO_c^M}$;

(6) $\overrightarrow{ihv_c(c)} = \overrightarrow{UO_c^M}$;

Return $\overrightarrow{ihv_c(c)}$;

3.5. XÂY DỰNG CẤU TRÚC IHV_ B$^+$Tree HỖ TRỢ TRUY VẤN KHOẢNG HIỆU QUẢ

Thuật toán xây dựng cây IHV_ B+Tree được mô tả như sau:

Thuật toán 3.4: ***Build_IHV_B$^+$Tree***

Input

Các giá trị trong **Bảng 3.2**; m: bậc của cây IHV_B^+Tree;

Output

$\boldsymbol{IHV\_B^+Tree}$

Các bước thực hiện:

(1) Xây dựng cây B$^+$Tree bậc m từ tập B là tập giá trị phân biệt của $V^B = \{v_1^B, v_2^B, \dots, v_z^B\}$, $B \subset V^B$.

(2) Mã hóa nút gốc và các nút trong của cây B$^+$Tree:

- Cấu trúc của nút gốc và nút trong có dạng: $(ptr\_node_0,\ k_1,\ ptr\_node_1,\ k_2,\ ptr\_node_2,\ k_3,\ \dots,\ ptr\_node_{m-1},\ k_m,\ ptr\_node_m)$ với $k_i \in B$ gọi là khóa (được sắp theo thứ tự $k_i < k_{i+1}, 1 \le i \le m-1$) và ptr_node_i là con trỏ tham chiếu tới cây con mà giá trị khóa ($keyvalue$) ở tất cả các nút của cây con đều thỏa mãn điều kiện $k_i \le keyvalue < k_{i+1}$.

- Để mã hoá nút gốc và nút trong của cây, ta thay các khóa k_i bằng vector giấu tin $\overrightarrow{ihv_b(k_i)}$ (chính là giá trị $\overrightarrow{ihv_b(v_i^B)}$ tương ứng). Khi đó cấu trúc nút gốc và nút trong có dạng: $(ptr\_node_0,\ \overrightarrow{ihv_b(k_1)},\ ptr\_node_1,\ \overrightarrow{ihv_b(k_2)},\ ptr\_node_2,\ \overrightarrow{ihv_b(k_3)},\ \dots,\ ptr\_node_{m-1},\ \overrightarrow{ihv_b(k_m)},\ ptr\_node_m)$.

(3) Cấu trúc lại và mã hóa các nút lá của cây B$^+$Tree:

Giả sử có n nút lá, khi đó cấu trúc mặc định của nút lá thứ i $(1 \le i \le n)$ trong cây B$^+$Tree có dạng: $(leaf\_previous_i,\ <k_{i(1)},\ ptr\_row_{i(1)}>,\ <k_{i(2)},\ ptr\_row_{i(2)}>,\ <k_{i(3)},\ ptr\_row_{i(3)}>\ \dots, <k_{i(m'-1)},\ ptr\_row_{i(m'-1)}>,\ <k_{i(m')},\ ptr\_row_{i(m')}>,\ leaf\_next_i)$ với $m' \le m$, $leaf\_previous_i$ và $leaf\_next_i$ lần lượt là các con trỏ lưu địa chỉ của các DiskBlock chứa nút lá thứ $i-1$ và nút lá thứ $i+1$. Còn $ptr\_row_{i(j)}$ $(1 \le j \le m')$ là các DataPointer, mỗi $ptr\_row_{i(j)}$ trỏ tới một danh sách liên kết lưu trữ các bản ghi được đại diện bởi khóa NewBucketIndex $k_{i(j)}$.

Để tái cấu trúc và mã hoá nút lá, thực hiện tuần tự như sau:

- *Thứ nhất:* biến đổi $<k_{i(j)},\ ptr\_row_{i(j)}>$ thành $<\overrightarrow{ihv_b(k_{i(j)})},\ ptr\_row_{i(j)}>$ với $k_{i(j)} \in B$.

- *Thứ hai:* $ptr\_row_{i(j)}$ trỏ tới đầu của danh sách liên kết $List_{i(j)} = \{row_1,\ row_2,\ \dots row_{i'}, \dots,\ row_{n'}\}$ với n' là số lượng bản ghi chứa NewBucketIndex $k_{i(j)}$. Mỗi phần tử trong danh sách liên kết là một bộ $row_{i'} = <row(r\_id),\ \overrightarrow{ihv_b(v_i)},\ ptr\_row\_next>$ với $row(r\_id)$ là một bản ghi trong quan hệ $R^E(ID, X_1, X_2, \dots X_t^E, \dots X_n)$. Giá trị $\overrightarrow{ihv_b(v_i)}$ là vector giấu tin cho một giá trị rõ v_i được đại diện bởi NewBucketIndex $k_{i(j)}$. Còn ptr_row_next là con trỏ lưu địa chỉ của bộ $row_{i'+1}$ tiếp theo, chú ý thứ tự của các phần tử $row_{i'}$ trong danh sách liên kết được sắp xếp ngẫu nhiên.

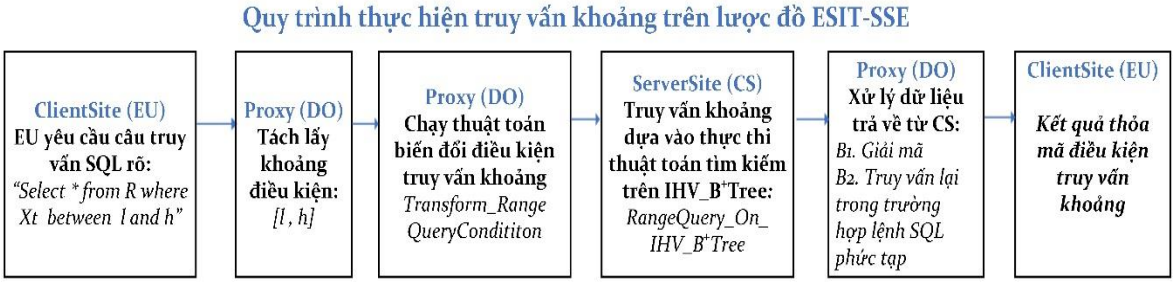
(4) Return IHV_B^+Tree

3.6. TRUY VẤN KHOẢNG TRONG LƯỢC ĐỒ ESIT-SSE TRIỂN KHAI TRÊN MÔ HÌNH DAS-PROXY

3.6.1. Quy trình truy vấn khoảng

Theo mô hình đề xuất trong **Hình 3.1**, quá trình thực hiện truy vấn khoảng diễn ra qua 4 bước (**Hình 3.4**), gồm: **Bước 1** – Biến đổi toán hạng so sánh trong biểu thức điều kiện truy vấn khoảng; **Bước 2** – Truy

vấn khoảng trên cấu trúc dữ liệu $\boldsymbol{IHV\_B^{+}Tree}$; **Bước 3** – Giải mã dữ liệu và truy vấn lại.



Hình 3.4. Quy trình truy vấn khoảng trên lược đồ ESIT-SSE

3.6.2. Thuật toán biến đổi khoảng truy vấn trên dữ liệu rõ sang các khoảng truy vấn trên IHV

Thuật toán 3.5: $\boldsymbol{Transform\_RangeQueryCondition}$

Input

l, h: các toán hạng so sánh của điều kiện truy vấn khoảng trên dữ liệu rõ;

Bảng các thông tin bí mật để sinh NewBucketIndex (**Bảng 3.1**);

Vector cơ sở $\overrightarrow{U_c}$ bí mật có chiều dài $k-2$; Cặp vị trí bí mật $(pos1, pos2) | \ 1 \le pos1, pos2 \le k$;

Ma trận bí mật M khả nghịch và có kích cỡ kxk;

Output

Các khoảng truy vấn trên vector giấu tin: $[\overrightarrow{ihv_c(l)}, \overrightarrow{ihv_c(h)}]$ và $[\overrightarrow{ihv_c(c_l)}, \overrightarrow{ihv_c(c_h)}]$;

Các bước thực hiện

Dựa vào "Bảng các thông tin bí mật để sinh NewBucketIndex", thực hiện :

(1) Xác định miền giá trị rõ thoả mãn điều kiện truy vấn khoảng $[l, h]$:

- Tìm $d_i.v$ lớn nhất mà nhỏ hơn hoặc bằng l, ký hiệu là v_l;
- Tìm $d_i.v$ nhỏ nhất mà lớn hơn hoặc bằng h, ký hiệu là v_h;

(2) Xác định l^B là danh sách các NewBucketIndex đại diện cho v_l, sau đó tìm $c_l = Min(l^B)$ là giá trị NewBucketIndex nhỏ nhất trong danh sách l^B;

(3) Xác định h^B là danh sách các NewBucketIndex đại diện cho v_h, sau đó tìm $c_h = Max(h^B)$ là giá trị NewBucketIndex lớn nhất trong danh sách h^B;

(4) Tính:

$\overrightarrow{UO_l^M} = \boldsymbol{Transform\_ComparisonOperand\_To\_IHV}(l, \overrightarrow{U_c}, pos1, pos2, M)$;

$\overrightarrow{UO_h^M} = \boldsymbol{Transform\_ComparisonOperand\_To\_IHV}(h, \overrightarrow{U_c}, pos1, pos2, M)$;

$\overrightarrow{UO_{c_l}^M} = \boldsymbol{Transform\_ComparisonOperand\_To\_IHV}(c_l, \overrightarrow{U_c}, pos1, pos2, M)$;

$\overrightarrow{UO_{c_h}^M} = \boldsymbol{Transform\_ComparisonOperand\_To\_IHV}(c_h, \overrightarrow{U_c}, pos1, pos2, M)$;

(5) Đặt:

$\overrightarrow{ihv_c(l)} = \overrightarrow{UO_l^M}$ và $\overrightarrow{ihv_c(h)} = \overrightarrow{UO_h^M}$; $\overrightarrow{ihv_c(c_l)} = \overrightarrow{UO_{c_l}^M}$ và $\overrightarrow{ihv_c(c_h)} = \overrightarrow{UO_{c_h}^M}$

(6) Return $[\overrightarrow{ihv_c(l)}, \overrightarrow{ihv_c(h)}]$ và $[\overrightarrow{ihv_c(c_l)}, \overrightarrow{ihv_c(c_h)}]$;

3.6.3. Thuật toán truy vấn khoảng trên cấu trúc dữ liệu $\boldsymbol{IHV\_B^{+}Tree}$

Thuật toán 3.6: $\boldsymbol{RangeQuery\_On\_IHV\_B^{+}Tree}$

Input

$IHV\_B^{+}Tree$; $R^E(ID, X_1, X_2, \ldots X_t^E, \ldots X_n)$;

$[\overrightarrow{ihv_c(l)}, \overrightarrow{ihv_c(h)}]$ và $[\overrightarrow{ihv_c(c_l)}, \overrightarrow{ihv_c(c_h)}]$;

Output

Tập bản ghi mã trong quan hệ $R^E(ID, X_1, X_2, \ldots X_t^E, \ldots X_n)$ thỏa mãn $l \le X_t \le h$;

Các bước thực hiện:

(1). Từ nút gốc của cây $\boldsymbol{IHV\_B^{+}Tree}$. So sánh $\overrightarrow{ihv_c(c_l)}$ với các khóa $\overrightarrow{ihv_b(k_l)}$ ở nút gốc $(ptr\_node_0, \overrightarrow{ihv_b(k_1)}, ptr\_node_1, \overrightarrow{ihv_b(k_2)}, ptr\_node_2, \ldots, \overrightarrow{ihv_b(k_m)}, ptr\_node_m)$;

(2). Nếu $\overrightarrow{ihv_b(k_1)}.\overrightarrow{vec(c_l)} > 0$ thì chuyển đến cây con bên trái của $\overrightarrow{ihv_b(k_1)}$;

(3). Nếu $\overrightarrow{ihv_b(k_1)}.\overrightarrow{ihv_c(c_l)} = 0$ thì thực hiện so sánh tiếp với $\overrightarrow{ihv_b(k_2)}$:

- Nếu $\overrightarrow{ihv_b(k_2)}.\overrightarrow{ihv_c(c_l)} > 0$ thì chuyển đến cây con bên trái của $\overrightarrow{ihv_b(k_2)}$;

- Nếu $\overrightarrow{ihv_b(k_2)}.\overrightarrow{ihv_c(c_l)} < 0$ thì tiếp tục so sánh với $\overrightarrow{ihv_b(k_3)}$, $\overrightarrow{ihv_b(k_4)}$, ... $\overrightarrow{ihv_b(k_m)}$ như ở bước **(2)** và bước **(3)**;

(4). Lặp lại các bước trên trong mỗi cây con cho đến khi duyệt tới nút lá;

(5). Giả sử sau khi kết thúc bước **(4)**, quá trình duyệt dừng ở node lá thứ i có cấu trúc $(leaf_{previous_i'} < \overrightarrow{ihv_b(k_{\iota(1)})}, ptr_{row\,i(1)} >, < \overrightarrow{ihv_b(k_{\iota(2)})}, ptr_{row\,i(2)} >, < \overrightarrow{ihv_b(k_{\iota(3)})}, ptr\_row_{i(3)} > \ldots, < \overrightarrow{ihv_b(k_{\iota(m'-1)})}, ptr\_row_{i(m'-1)} >, < \overrightarrow{ihv_b(k_{\iota(m')})}, ptr\_row_{i(m')} >, leaf\_next_i)$. Khi đó:

(5.1). Bắt đầu từ bộ $< \overrightarrow{ihv_b(k_{\iota(1)})}, ptr\_row_{i(1)} >$, nếu $\overrightarrow{ihv_b(k_{\iota(1)})}.\overrightarrow{ihv_c(c_l)} \geq 0\ and\ \overrightarrow{ihv_b(k_{\iota(1)})}.\overrightarrow{ihv_c(c_h)} \leq 0$ thì thực hiện tiếp bước **(5.2)**, ngược lại dừng bước **(5)**;

(5.2). Xét bộ $< \overrightarrow{ihv_b(k_{\iota(1)})}, ptr\_row_{i(1)} >$, truy xuất vào danh sách liên kết $List_{i(1)} = \{row_1, row_2, \ldots row_{i'}, \ldots, row_{n'}\}$ được trỏ bởi $ptr\_row_{i(1)}$;

(5.3). Duyệt tuần tự từng phần tử của danh sách liên kết $List_{i(1)}$. Với mỗi phần tử $row_{i'} = < row(ID), \overrightarrow{ihv_b(v_l)}, ptr\_row\_next >$ kiểm tra:

- Nếu $\overrightarrow{ihv_b(v_l)}.\overrightarrow{ihv_c(l)} \geq 0\ and\ \overrightarrow{ihv_b(v_l)}.\overrightarrow{ihv_c(h)} \leq 0$ thì trả về giá trị $row(ID)$;
- Ngược lại thì bỏ qua $row_{i'}$ và tiếp tục lặp lại quá trình kiểm tra ở trên với các phần tử kế tiếp trong danh sách cho đến khi $i' = n'$;

(5.4). Lặp lại bước **(5.1)** đến bước **(5.3)** đối với các bộ $< k_{i(2)}, ptr\_row_{i(2)} >$, $< k_{i(3)}, ptr\_row_{i(3)} > \ldots < k_{i(m')}, ptr\_row_{i(m')} >$;

(5.6). Lặp lại bước **(5)** đối với các node lá thứ $i+1,\ i+2, \ldots,\ n$;

(6). Tham chiếu các $row(ID)$ tìm được vào R^E để tìm ra các bản ghi mã thỏa mãn $l \leq X_t \leq h$;

(7). Trả về tập các bản ghi mã thỏa mãn $\boldsymbol{l \leq X_t \leq h}$;

3.7. PHÂN TÍCH BẢO MẬT CỦA LƯỢC ĐỒ ESIT-SSE

Với các đặc tính tương đồng trong thiết kế của lược đồ đề xuất với những lược đồ trước đó thì ESIT-SSE được coi đáp ứng mô hình bảo mật IND-CKA2. Sau đây là một số phân tích bổ sung ESIT-SSE.

3.7.1. Vấn đề rò rỉ thông tin tĩnh và động

Rò rỉ thông tin tĩnh được hiểu là kẻ gian (gọi tắt là A) khai thác được các thông tin dựa trên những kiến thức đã có về bản rõ và bản mã. Rò rỉ thông tin động được A khai thác thông qua việc quan sát cấu trúc mẫu truy vấn và tập kết quả trả về kết hợp cùng một số kiến thức biết trước về bản rõ, bản mã.

3.7.2. Phân tích bảo mật NewBucketIndex

Với cách xây dựng Bucket bảo toàn thứ tự và không nạp chồng, sẽ rất khó để đảm bảo các giá trị về phương sai (σ^2), entropy (H) và độ lệch phân phối xác suất (pvd) đồng thời tối ưu được. Giải pháp xây dựng Bucket đạt độ bảo mật cao nhất phải là nạp chồng giá trị giữa các Bucket và không bảo toàn thứ tự. Tuy nhiên, thuật toán xây dựng rất phức tạp, hiệu năng truy vấn thấp và không hỗ trợ truy vấn khoảng do không bảo toàn được thứ tự các Bucket. Vì vậy, các NewBucketIndex được thiết kế nạp chồng nhưng vẫn bảo toàn thứ tự để cố gắng nâng cao khả năng bảo mật nhưng vẫn giữ được lợi thế trong biến đổi điều kiện truy vấn khoảng và thực thi tìm kiếm trên các cấu trúc phát triển từ cây B^+Tree. Luận án thử nghiệm các giá trị σ^2, H, pvd theo hai phương pháp tạo khác nhau: ***Phương pháp 1***. Tạo 3 Buckets tuần tự và không nạp chồng; ***Phương pháp 2.*** Sử dụng thuật toán $BuildNewBucketIndex$ tạo 6 Buckets tuần tự và nạp chồng

Bảng 3.1. So sánh $\boldsymbol{\sigma^2, H, pvd}$ giữa hai phương pháp tạo Bucket

Bucket	$\boldsymbol{\sigma^2}$	$\boldsymbol{H}$	$\boldsymbol{pvd}$
$B1$	4.5	0.986	32
$B2$	7.5	0.971	50
$B3$	148	0.869	1250
$NB1$	81	0.811	648
$NB2$	5.3	0.914	32

$NB3$	8.9	1.5	389
$NB4$	36.3	0.971	242
$NB5$	208.3	0.915	1250
$NB6$	1682	1	6728

Từ **Bảng 3.1**, có thể thấy hầu hết các giá trị $\sigma^2,\ pvd$ đo được trên các Bucket tạo bởi phương pháp 2 đều có giá trị cao hơn so với phương pháp 1.

3.7.3. Phân tích bảo mật vector giấu tin IHV

Một số phương pháp phân tích mà kẻ gian thường sử dụng để khai thác thông tin từ IHVgồm:

- Thống kê tần suất xuất hiện của các vector $(\overrightarrow{UO_b}, \overrightarrow{UO_c})$ để đoán vector nào xuất hiện nhiều nhất hoặc ít nhất, qua đó suy luận được giá trị b, c tương ứng. Tuy nhiên do tác động của cặp số ngẫu nhiên $(r_b,\ r_c)$ nên cùng một giá trị rõ b hoặc c sẽ có nhiều trạng thái vector $\overrightarrow{UO_b}$, $\overrightarrow{UO_c}$ khác nhau. Cách này không khả thi.

- A phân tích nội dung trong các vector nhúng $(\overrightarrow{UO_b}, \overrightarrow{UO_c})$ và cố gắng dự đoán cặp vector trực giao ($\overrightarrow{U_b}$, $\overrightarrow{U_c}$) có chiều dài $k-2$. Bằng phương pháp tìm ước chung lớn nhất cố gắng tìm cặp vị trí $(pos1, pos2)$. Để tìm được 2 vị trí trong k vị trí, ta cần tới k$(k-1)/2$ phép thử. Vì vậy, luận án đề xuất sử dụng thêm cặp ma trận bí mật (M_b, M_c) để biến đổi $\overrightarrow{UO_b}, \overrightarrow{UO_c}$ thành các vector giấu tin $\overrightarrow{UO_b^M}, \overrightarrow{UO_c^M}$ bảo mật hơn.

Để phân tích khả năng chống rò rỉ thông tin từ $\overrightarrow{UO_b^M}, \overrightarrow{UO_c^M}$, luận án xét giả thuyết tấn công biết trước bản rõ. Khi đó A biết trước tập vector $(\overrightarrow{UO_b}, \overrightarrow{UO_c})$ và tập vector $(\overrightarrow{UO_b^M}, \overrightarrow{UO_c^M})$ tương ứng. Mục tiêu cần tìm ra khoá là ma trận bí mật M khả nghịch cỡ kxk. Việc tìm M tương đương tìm M_b trong phương trình $UO_b^M = M_b$x UO_b. Với UO_b^M, UO_b là các ma trận cỡ $kx1$ được chuyển đổi từ vector $\overrightarrow{UO_b^M}$ và $\overrightarrow{UO_b}$. Có vô số ma trận M_b thoả mãn phương trình, với mỗi cặp (UO_b^M, UO_b) được quan sát, kẻ gian cũng nhận được vô số M_b khác nhau. Rõ ràng việc tìm ra M chính xác (lưu trữ tại Proxy) từ vô số các M_b nói trên là một thách thức lớn.

3.7.4. Phân tích bảo mật cây IHV_B$^+$Tree

Cấu trúc của cây IHV_B^+Tree được xây dựng theo thuật toán $Build\_IHV\_B^+Tree$ đáp ứng việc chống rò rỉ thông tin trên CS khi kẻ gian cố gắng khai thác mối quan hệ giữa các khóa trong mỗi nút và giữa các nút. Cụ thể: Mỗi khóa (NewBucketIndex) trong các nút của cây có thể đại diện cho nhiều giá trị rõ v_i khác nhau và ngược lại. Thứ tự tăng dần của các khoá lân cận nhau trong mỗi nút sẽ không phản ánh hoàn toàn sự chính xác về thứ tự giữa các giá trị rõ v_i mà khóa đó đại diện. Tương tự, mối quan hệ về độ lớn của khoá giữa nút cha và nút con hoặc giữa các nút lá với nhau cũng không phản ánh chính xác việc thứ tự các giá trị rõ v_i mà các khoá đó đại diện. Nếu kẻ gian dựa trên thứ tự khóa trên các nút lá để suy diễn ra thứ tự các bản ghi mã thì không có căn cứ bởi hai lý do: thứ nhất các bản ghi cùng chứa một giá trị v_i hoàn toàn có thể nằm trên các danh sách liên kết của các nút lá khác nhau; thứ hai thứ tự các bản ghi trong danh sách liên kết tại một nút lá được sắp xếp ngẫu nhiên.

3.8. THỰC NGHIỆM VÀ ĐÁNH GIÁ HIỆU NĂNG CỦA LƯỢC ĐỒ ESIT-SSE

3.8.1. Chuẩn bị điều kiện thực nghiệm

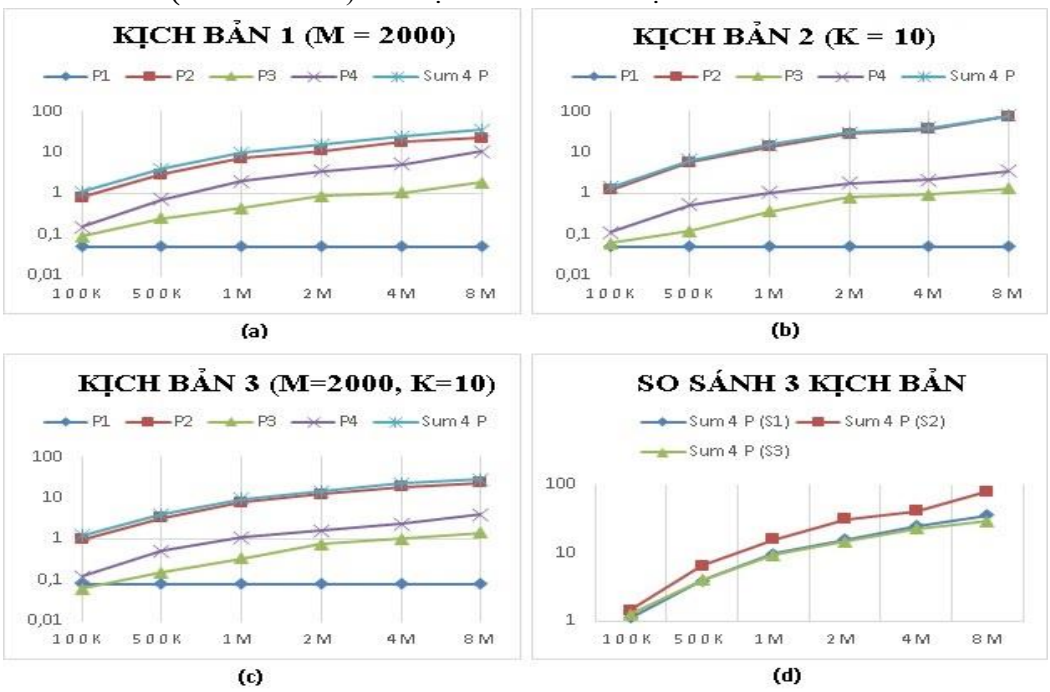
Thời gian truy vấn: được đo dựa trên bốn pha: ***Pha 1 (P1):*** gồm tổng thời gian gửi câu lệnh truy vấn từ ClientSite lên Proxy, thời gian biến đổi điều kiện truy vấn trên Proxy và thời gian gửi điều kiện truy vấn sau khi biến đổi đến CS; ***Pha (P2):*** thời gian truy vấn và trả về tập kết quả dữ liệu mã trên CS; ***Pha (P3):*** thời gian gửi tập kết quả dữ liệu mã từ CS về Proxy; ***Pha 4 (P4):*** thời gian giải mã tại Proxy và thời gian trả kết quả cuối cùng về ClientSite. **Sum4P:** tổng thời gian thực hiện của 4 pha (P1+P2+P3+P4).

Kịch bản thử nghiệm: Sử dụng 3 kịch bản, gồm: ***Kịch bản 1.*** Xây dựng cấy B+Tree cho các NewBucketIndex và truy vấn khoảng trực tiếp qua việc duyệt cây này; ***Kịch bản 2.*** Truy vấn trực tiếp qua so sánh với các giá trị IHV tại các bản ghi dữ liệu; ***Kịch bản 3.*** Truy vấn qua giải pháp của lược đồ ESIT-SSE.

Dữ liệu thử nghiệm: Luận án sử dụng dữ liệu cột L_EXTENDEDPRICE của bảng LINEITEM trong CSDLQH TPC-H cho các kịch bản thử nghiệm. Thực thi công cụ DBGen để sinh lần lượt 100 ngàn, 500 ngàn, 1 triệu, 2 triệu, 4 triệu và 8 triệu bản ghi cho LINEITEM. Xây dựng CSDLQH mã TPC_H_CS trong đó có bảng LINEITEM_CS với trường dữ liệu mã hóa L_EXTENDEDPRICE_E.

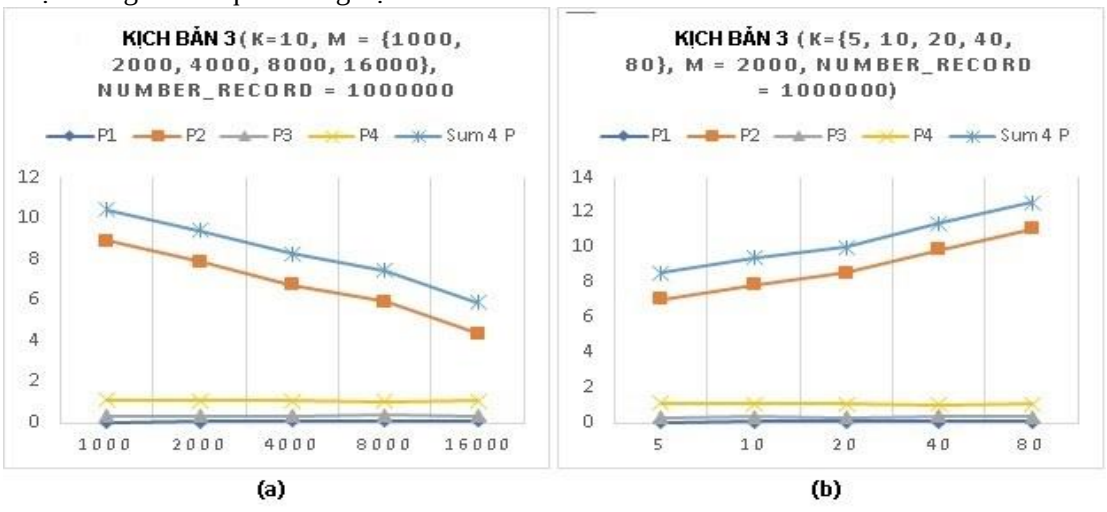
3.8.2. Thực nghiệm và đánh giá hiệu năng tìm kiếm

Luận án chọn tham số $m = 2000$ (số lượng NewBucketIndex) cho Kịch bản thứ nhất và Kịch bản thứ ba. Chọn tham số $k = 10$ (chiều dài IHV) cho Kịch bản thứ 2 và Kịch bản thứ ba. Kết như **Hình 3.5.**



Hình 3.5. Đánh giá thời gian thực thi 4 pha trên 3 kịch bản thử nghiệm

Luận án tiếp tục thử nghiệm Kịch bản 3 bằng việc cho biến động một trong hai tham số m và k khi truy vấn trên 1 triệu bản ghi. Kết quả thử nghiệm như **Hình 3.6**.



Hình 3.6. Đánh giá thời gian thực thi lược đồ ESIT-SSE khi thay đổi k, m

3.9. So sánh hiệu quả lược đồ ESIT-SSE với một số nghiên cứu

Trong phần này luận án tiếp tục làm rõ hơn các đánh giá về tính hiệu quả giữa các giải pháp trên góc nhìn tổng hợp và đối sánh:

Đánh giá chung: Các công trình đã liệt kê đều tiêu biểu và liên quan tới vấn đề truy vấn khoảng trên

dữ liệu số mã hóa, được đăng trên các nhà xuất bản/tạp chí uy tín, trong đó nhiều tài liệu cập nhật được tình hình trong 5 năm trở lại đây; Cấu trúc dữ liệu hỗ trợ xây dựng chỉ mục trong lược đồ ESIT-SSE có tính mới so với các công trình liên quan khi đề xuất và sử dụng kết hợp NewBucketIndex, IHV và cây IHV_B^+Tree.

Đánh giá về lược đồ ESIT-SSE khi so với các công trình liên quan: Các thuật toán Transform_ComparisonOperand_To_IHV và Transform_RangeQueryCondition đều khai thác dữ liệu từ Bảng 3.1 nên lược đồ ESIT -SSE không tốn nhiều chi phí vào quá trình giao tiếp cũng như tính toán tại Proxy hay Client. Chi phí tìm kiếm chủ yếu tập trung tại CS qua việc thực thi thuật toán RangeQuery_On_IHV_B$^+$Tree. Trong đó, quá trình duyệt cây đến các nút lá có chi phí tương tự duyệt cây B+Tree với độ phức tạp trung bình $O(\log N_b)$, quá trình duyệt các LIST trên các nút lá vừa tìm kiếm được có chi phí cỡ $O(N_{br})$. Như vậy với tổng chi phí tìm kiếm tại Server cỡ $O(\log N_b + N_{br})$ được đánh giá là rất tốt khi so sánh với các công trình khác trong bảng. Xét về số vòng giao tiếp dữ liệu giữa EU – PROXY - CS khi truy vấn thì ESIT-SSE cũng cho thấy ưu thế khi chỉ tốn 1 vòng. Về khả năng chống rò rỉ chỉ mục và mẫu truy vấn cũng đạt yêu cầu. Qua đó thêm một lần nữa chỉ ra tính hiệu quả của lược đồ ESIT-SSE.

3.10. KẾT LUẬN CHƯƠNG 3

Chương này đề xuất lược đồ ESIT-SSE và quy trình triển khai trên mô hình DAS-PROXY cho phép truy vấn khoảng hiệu quả dữ liệu số trong CSDLQH mã. Theo đó nội dung trọng tâm là xây dựng chỉ mục NewBucketIndex, vector giấu tin IHV và cấu trúc dữ liệu IHV_B+ Tree nhằm tạo ra giải pháp truy vấn khoảng có nhiều ưu điểm trên các tiêu chí về: bảo mật, hiệu năng và khả thi triển khai. Cụ thể:

- Các NewBucketIndex được xây dựng đảm bảo nguyên tắc nạp chồng nhưng vẫn bảo toàn thứ tự. Điều này cho phép các NewBucketIndex tăng cường bảo mật, hỗ trợ tăng tốc truy vấn qua các cấu trúc như B+Tree và giảm bớt bản ghi dương tính giả trả về. Tuy nhiên tính chất bảo toàn thứ tự vẫn gây rò rỉ thông tin cho NewBucketIndex. Để giải quyết vấn đề này, các NewBucketIndex tiếp tục được biến đổi thành vector giấu tin IHV với khả năng che giấu thứ tự nhưng vẫn hỗ trợ cơ chế so sánh bảo mật. Để tăng hiệu quả tìm kiếm dựa trên tính chất bảo toàn thứ tự của NewBucketIndex và bảo mật cao của IHV, cấu trúc IHV_B+ Tree được xây dựng cho phép truy vấn khoảng thông qua thao tác duyệt cây tốc độ cao thay vì thực hiện quét toàn bộ các bản ghi trong bảng của CSDLQH mã.

- Tính khả thi được kế thừa từ các lợi thế triển khai của mô hình DAS-PROXY, hỗ trợ truy vấn với một số lệnh mở rộng và cho phép thay đổi các tham số tác động liên quan. Trong khi đó, tính bảo mật (khả năng chống rò rỉ thông tin chỉ mục, mẫu truy vấn) và hiệu năng (kết quả trả về không dư thừa, thời gian thực thi, độ phức tạp tính toán, số vòng giao tiếp) được làm rõ qua các kịch bản tấn công, kịch bản thực nghiệm và bảng so sánh với các công trình tiêu biểu liên quan.

Bên cạnh các ưu điểm đã nêu, thì lược đồ ESIT-SSE cũng còn tồn tại một số hạn chế như sau:

- Số lượng m Bucket càng lớn thì độ bảo mật của NewBucketIndex càng giảm và ngược lại. Nếu điều chỉnh m nhỏ, một NewBucketIndex sẽ đại diện cho nhiều giá trị bản rõ hơn và kéo theo số phần tử của mỗi $List_{i(j)}$ gán với từng nút lá (trong Thuật toán 3.4: Build_IHV_B$^+$Tree) tăng lên làm chi phí duyệt cây tốn kém hơn.

- Các phần tử trong mỗi danh sách $List_{i(j)}$ được sắp xếp ngẫu nhiên nhằm tăng cường tính bảo mật, nhưng lại làm thao tác duyệt tìm các phần tử $\overrightarrow{ihv_b(v_1)}$ trong danh sách mà thoả mãn điều kiện truy vấn khoảng sẽ phức tạp hơn so với một danh sách đã được sắp xếp.

- Tham số k càng lớn càng tăng khả năng chống rò rỉ của vector IHV, tuy nhiên kéo theo chi phí lưu trữ vector IHV cũng tăng cao và chi phí duyệt cây cũng bị ảnh hưởng.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

A. Các kết quả đạt được của luận án

1) Áp dụng thành công mô hình DAS-PROXY để triển khai thống nhất cho hai lược đồ DIQ-SSE và ESIT-SSE đề xuất trong luận án. Máy chủ Proxy trong mô hình cho phép: quản lý và lưu trữ an toàn các khóa/metadata/thuật toán bí mật; giảm tải cho thiết bị cuối của EU; và hỗ trợ các vấn đề mở rộng liên quan tới vận hành. Mô hình DAS-PROXY đảm bảo tiêu chí "tính khả thi trong triển khai" khi xét tới hiệu quả của các lược đồ đề xuất trong luận án.

2) Đề xuất lược đồ DIQ-SSE hỗ trợ truy vấn chuỗi con hiệu quả trong CSDLQH mã. Các điểm đóng góp gồm: đề xuất các cấu trúc dữ liệu mới như "Tập các cặp ký tự kết nối phân biệt theo khoảng cách trên chuỗi rõ" hay "Tập các mảng nhị phân biểu diễn vị trí ký tự của chuỗi rõ" để xây dựng cặp chỉ mục $Index1, Index2$ hỗ trợ tìm kiếm; đề xuất quy trình truy vấn hiệu quả và các thuật toán liên quan trên cặp chỉ mục $Index1, Index2$.

Tính hiệu quả của lược đồ DIQ-SSE được đánh giá qua phân tích các kịch bản tấn công, kịch bản thực nghiệm trên số lượng bản ghi lớn của CSDL TPC-H cũng như qua so sánh tương quan với các công trình tiêu biểu liên quan. Về bảo mật: các chỉ mục và mẫu truy vấn có khả năng chống rò rỉ thông tin bản rõ; Về hiệu năng: ghi nhận ưu điểm về tỷ lệ lọc, tỷ lệ lỗi và thời gian thực thi truy vấn và quy trình truy vấn tuần tự tận dụng được lợi thế của các chỉ mục. Bên cạnh đó, quá trình truy vấn chỉ sử dụng 1 vòng giao tiếp dữ liệu từ EU đến CS với độ phức tạp tìm kiếm cỡ $O(k)$ trên $Index1$ và $O(2.l_{substr})$ trên $Index2$. Về hỗ trợ dạng thức truy vấn phổ biến: cho phép thực thi chuỗi "$LIKE\ '\%\ substring\ \%'$". Ngoài ra lược đồ cho phép điều chỉnh các tham số $u,\ m$ tác động đến hiệu năng, bảo mật và không gian lưu trữ, qua đó thể hiện tính khả thi của lược đồ DIQ-SSE trong triển khai thực tiễn.

3) Đề xuất lược đồ ESIT-SSE hỗ trợ truy vấn khoảng hiệu quả trên dữ liệu số trong CSDLQH mã. Các điểm mới trong nội dung đề xuất gồm: xây dựng chỉ mục NewBucketIndex, vector giấu tin (IHV) và cây IHV_B+ Tree; xây dựng quy trình và các thuật toán hỗ trợ truy vấn khoảng trên cây IHV_B+ Tree.

Tính hiệu quả của lược đồ ESIT-SSE được đánh giá qua phân tích các kịch bản tấn công, kịch bản thực nghiệm trên số lượng bản ghi lớn của CSDL TPC-H cũng như qua so sánh tương quan với các công trình tiêu biểu liên quan. Về bảo mật: các chỉ mục NewBucketIndex, vector IHV, cây IHV_B+ Tree và mẫu truy vấn có khả năng chống rò rỉ thông tin bản rõ; Về hiệu năng: hỗ trợ 1 vòng giao tiếp duy nhất giữa EU với CS; thuật toán duyệt cây IHV_B+ Tree tốc độ cao (độ phức tạp cỡ $O(logN_b + N_{br})$) giúp giảm đáng kể chi phí thời gian truy vấn. Về hỗ trợ dạng thức truy vấn phổ biến: cho phép thực thi chuỗi "$\boldsymbol{X_t}\ Between\ \boldsymbol{l}\ And\ \boldsymbol{h}$". Ngoài ra lược đồ cho phép điều chỉnh các tham số $k,\ m$ tác động đến hiệu năng, bảo mật và không gian lưu trữ, qua đó thể hiện tính khả thi của lược đồ ESIT-SSE trong triển khai thực tiễn.

B. Hướng phát triển của luận án

Ngoài các đóng góp đã được trình bày, NCS nhận thấy còn một số tồn tại trong luận án cùng các vấn đề mở rộng khác cũng nên được quan tâm giải quyết giúp hoàn thiện hơn các lược đồ đã đề xuất. Cụ thể:

- Nghiên cứu tối ưu chi phí không gian lưu trữ chỉ mục.
- Nghiên cứu đảm bảo chống tấn công mẫu truy cập trên CS (Access pattern).
- Nghiên cứu tiếp tục đa dạng hóa thêm các điều kiện truy vấn.
- Nghiên cứu các vấn đề mở rộng khác trong triển khai vận hành các lược đồ như: đổi khóa, biến động dữ liệu, cân bằng tải, …

CÔNG TRÌNH NGHIÊN CỨU CỦA TÁC GIẢ

[CT1] **C. Ngoc Hoang**, M. Hieu Nguyen, T. Thu Thi Nguyen, and H. Quang Vu, "A novel method for designing indexes to support efficient substring queries on encrypted databases," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 3, pp. 20–36, Mar. 2023, doi: 10.1016/j.jksuci.2023.02.008. **(*Journal: Scopus Index Q1, SCIE IF 5.2*)**

[CT2] **Hoàng Ngọc Cảnh**, Nguyễn Hiếu Minh, "Truy vấn hiệu quả dữ liệu ký tự mã hóa trong cơ sở dữ liệu thuê ngoài", *Kỷ yếu Hội nghị khoa học công nghệ quốc gia lần thứ XV – Nghiên cứu cơ bản và Ứng dụng công nghệ thông tin (FAIR 2022)*, ISBN: 978-604-357-119-6, Trang 8-15.

[CT3] **Hoang, N.C.**, Nguyen, T.T.T. (2024). "Build Feature Vector for String Supporting Pattern Matching on Encrypted Character Data". *In: Nguyen, T.D.L., Dawson, M., Ngoc, L.A., Lam, K.Y. (eds) Proceedings of the International Conference on Intelligent Systems and Networks. ICISN 2024.* Lecture Notes in Networks and Systems, vol 1077. Springer, Singapore. https://doi.org/10.1007/978-981-97-5504-2_41. **(*Proceedings: Scopus Index Q4*)**

[CT4] **Hoang, N.C.**, Nguyen, T.T.T., Tran, L.K.D., Vu, Q.H. (2024). "A Secure Approach to Financial Data Management: A Method for Constructing Encrypted Indexes to Support Efficient Querying without Revealing Order Information". *Hanoi University of Industry Journal of Science and Technology (P-ISSN 1859-3585, E-ISSN 2615-9619)*, vol.60, no.11E (Nov 2024), doi: http://doi.org/10.57001/huih5804.2024.344.

[CT5] **Hoang, N.C.**, Nguyen, T.T.T., Tran, L.K.D., Vu, Q.H. (2024). "Design Index Supporting Efficient Querying On Encrypted Numerical Data Based On Stacked Bucket Loading and Order Preservation". *In: Proceedings of the International Conference on Applied Mathematics and Computer Science (ICAMCS2024).* Lecture Notes in Networks and Systems, vol 1, Springer, Singapore. **(*Proceedings: Scopus Index Q4*)**