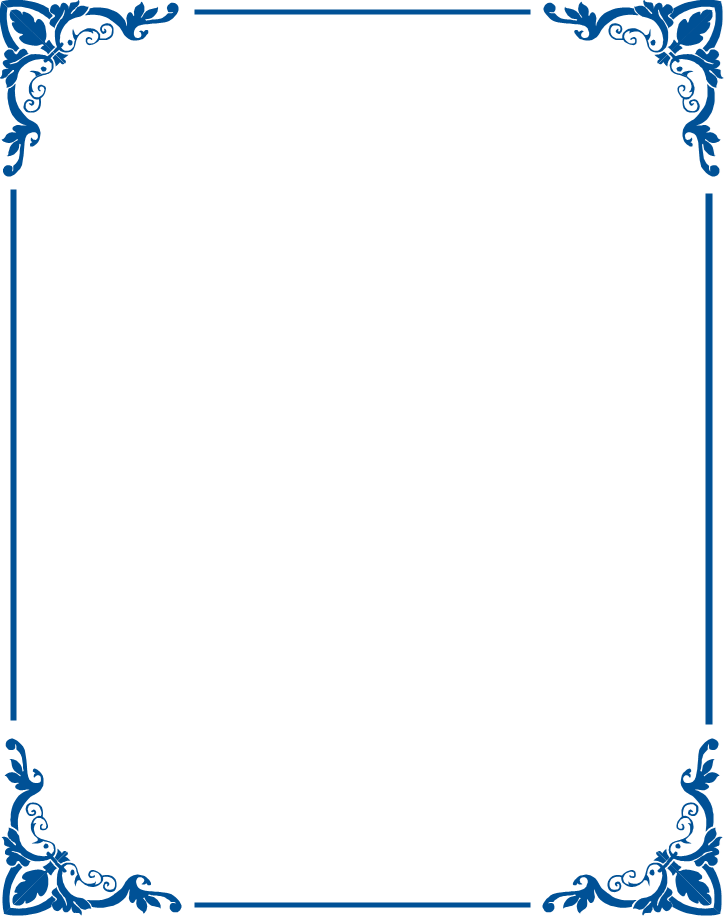
****

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**MÔN: MẬT MÃ HỌC**

**ĐỀ TÀI: MULTI‑SERVER SEARCHABLE DATA CRYPT**

**Giảng viên hướng dẫn:** *Nguyễn Ngọc Tự*

**Sinh viên thực hiện:** *Nhóm 11*

*Nguyễn Ngọc Trưởng 19522440*

*Hồ Thị Ngọc Phúc 19520220*

**MỤC LỤC**

[**Danh mục hình ảnh** 3](#_Toc76567230)

[**Danh mục bảng** 3](#_Toc76567231)

[**I.** **Ngữ cảnh** 4](#_Toc76567232)

[**II.** **Mục tiêu đề tài** 5](#_Toc76567233)

[**III.** **Research motivations** 5](#_Toc76567234)

[**IV.** **Searchable encryption** 6](#_Toc76567235)

[**V.** **Hệ thống mã hóa tìm kiếm dữ liệu đa máy chủ (MS-SDC) được đề xuất** 7](#_Toc76567236)

[**1.** **Kế hoạch và giải pháp** 7](#_Toc76567237)

[**2.** **MS-SDC Interface** 7](#_Toc76567238)

[**3.** **MS-SDC Manager** 7](#_Toc76567239)

[**4.** **MS-SDC API** 8](#_Toc76567240)

[*a)* *Chức năng tải lên* 8](#_Toc76567241)

[b) *Chức năng tìm kiếm* 8](#_Toc76567242)

[*c)* *Chức năng tải xuống* 9](#_Toc76567243)

[**5.** **Chi tiết các thuật toán của hệ thống được đề xuất** 9](#_Toc76567244)

[*a)* *FileUpload\_Setup* 9](#_Toc76567245)

[*b)* *KeywordExtraction* 11](#_Toc76567246)

[*c)* *SearchEngine\_ Keyword* 11](#_Toc76567247)

[*d)* *FileDownload* 12](#_Toc76567248)

[**VI.** **Contributions** 14](#_Toc76567249)

[**VII. Comments** 14](#_Toc76567250)

[**VIII.** **Demo** 14](#_Toc76567251)

[**1.** **Triển khai:** 14](#_Toc76567252)

[**2.** **Demo** 15](#_Toc76567253)

[***TÀI LIỆU THAM KHẢO*** 20](#_Toc76567254)

[***NHẬN XÉT CỦA GIẢNG VIÊN*** 21](#_Toc76567255)

**Danh mục hình ảnh**

[Figure 1 Outsourcing system (Trích trong bài giảng MMH chương 7) 4](#_Toc76567326)

[Figure 2 Mô hình SSE (trích trong bài báo tham khảo (3)) 6](#_Toc76567327)

[Figure 3 Sơ đồ MS-SDC (trích bài báo tham khảo (2)) 7](https://uithcm-my.sharepoint.com/personal/19520220_ms_uit_edu_vn/Documents/FINAL_PROJECT_MMH.docx#_Toc76567328)

[Figure 4 Sơ đồ thuật toán Upload và KeyExtraction (trích bài báo tham khảo (2)) 10](#_Toc76567329)

[Figure 5 Sơ đồ chi tiết thuật toán Search (trích bài báo (2)) 12](#_Toc76567330)

[Figure 6 Sơ đồ chi tiết thuật toán Download (trích bài báo tham khảo (2)) 13](#_Toc76567331)

[Figure 7 Giao diện chương trình khi gõ lệnh --help 15](#_Toc76567332)

[Figure 8 Giao diện chương trình khi thực hiện uploadfile 16](#_Toc76567333)

[Figure 9 Query Keyword Table 17](#_Toc76567334)

[Figure 10 Index Entry Table 17](#_Toc76567335)

[Figure 11 Server List 18](#_Toc76567336)

[Figure 12 Giao diện chương trình khi thực hiện Search 19](#_Toc76567337)

[Figure 13 Giao diện chương trình khi thực hiện Download 19](#_Toc76567338)

[Figure 14 File headline.json đã được download trong thư mục Download 19](#_Toc76567339)

**Danh mục bảng**

[Bảng 1 Ví dụ về Index Entry Table 11](#_Toc76080116)

[Bảng 2 Ví dụ về Query Keyword Table 11](#_Toc76080117)

1. **Ngữ cảnh**

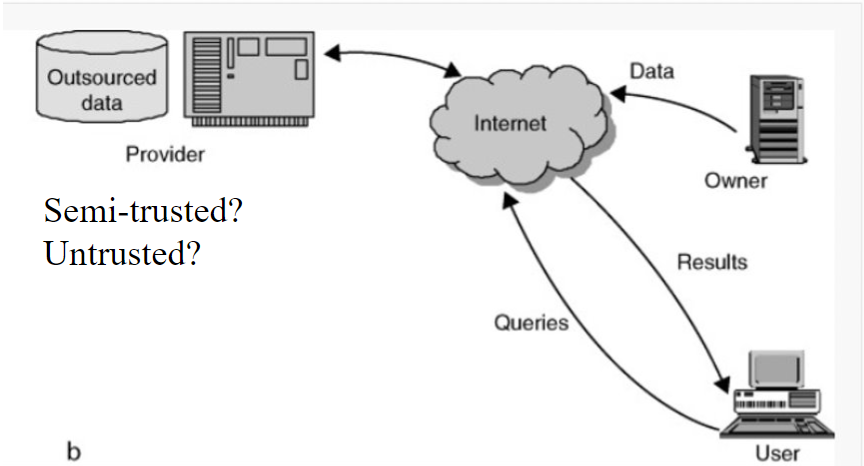
* Ngày nay, với sự bùng nổ của cách mạng công nghiệp 4.0, sự phát triển cao của công nghệ trong cả phần cứng lẫn phần mềm đã dẫn tới đến lượng dữ liệu cần lưu trữ ngày càng khổng lồ với các tổ chức lớn. Cho nên, có rất nhiều tổ chức, cá nhân, các tập đoàn đã sử dụng **outsourcing system,** lưu dữ liệu của mình ở 1 bên thứ 3 hỗ trợ dịch vụ này.

Figure 1 Outsourcing system (Trích trong bài giảng MMH chương 7)

* Theo một báo cáo của công ty kiểm toán bảo mật Netwrix, số lượng áp dụng đám mây trong các tổ chức được khảo sát đã tăng từ 43% (2015) lên 68% (2016) [1]. Báo cáo cũng nói rằng đám mây bảo mật là mối quan tâm hàng đầu đối với 70% tổ chức trên toàn thế giới. Trong một báo cáo được công bố vào năm 2016, Cisco dự đoán rằng vào năm 2020, 59% người dùng Internet trên thế giới sẽ sử dụng lưu trữ đám mây cá nhân, tăng từ 47% vào năm 2015. Những xu hướng này không phải là hiếm vì dịch vụ lưu trữ đám mây mang lại nhiều lợi ích cho người dùng. Tuy nhiên, dịch vụ này không phải là không có nhược điểm của nó.
* Các nhà cung cấp dịch vụ lưu trữ thường là semi-trusted và untrusted. Do đó, không có một đảm bảo nào cho dữ liệu của người dùng khi gửi qua cho các nhà cung cấp này sẽ được bảo vệ an toàn khỏi những kẻ tấn công cũng như các nhà quản trị viên, những người làm chủ dịch vụ. Vậy thì làm sao để có thể giữ an toàn cho dữ liệu của chính mình?

1. **Mục tiêu đề tài**

* Để tránh bị lộ dữ liệu, giải pháp là mã hoá hết tất cả dữ liệu trước khi gửi qua cho nhà cung cấp. Như vậy thì nhà cung cấp không thể hiểu được dữ liệu. Nhưng khi muốn sửa đổi 1 tài liệu nào đó thì làm sao để truy cập trong khi tất cả đều đã được mã hoá? Buộc người dùng phải tải toàn bộ tài liệu về, giải mã sau đó sửa đổi. Cách này rất tốn thời gian cũng như không hiệu quả, và thiếu thực tế.
* Cho nên chúng tôi muốn xây dựng một hệ thống multi-server searchable data crypt, hệ thống mã hoá và phân tán dữ liệu đến nhiều server khác nhau và hỗ trợ thuật toán searchable data crypt cho việc tìm kiếm dữ liệu trên các tài liệu đã được mã hoá. Từ đây, dữ liệu của người dùng có thể được đảm bảo an toàn do đã được mã hoá và thuận tiện vì được hỗ trợ tìm kiếm, download.

1. **Research motivations**

Hầu hết các sơ đồ được nghiên cứu trước Poh, 2017 làm việc dựa trên một máy chủ duy nhất, trong Poh, 2017 đề xuất làm việc trên 2 hay nhiều server. Tuy nhiên hầu hết các nghiên cứu trước đây đều nhiều hạn chế, một số thì phải chịu những hạn chế khi sử dụng một máy chủ duy nhất, một số thì chỉ xử lý hai máy chủ và giới thiệu các tính năng mới, tuy nhiên hạn chế với một số vấn đề bảo mật. Do vậy đây là động lực để phát triển một hệ thống có thể tăng cường hầu hết các vấn đề nói trên trong nỗ lực vừa có được nhiều tính năng hơn xử lý các hạn chế của các chương trình trước đó, bên cạnh việc duy trì giao diện người dùng đơn giản với bảo mật cao chống hack. Hệ thống sẽ có các đặc điểm sau:

* Hệ thống hoạt động trên đa server.
* Hệ thống sẽ chia nhỏ file của người dùng thành các block nhỏ, mã hoá các khối và gửi ngẫu nhiên đến các server, để không server nào có đầy đủ 1 bản file hoàn chỉnh.
* Hệ thống hỗ trợ tìm kiếm dựa trên từ khoá cho người dùng, người dùng có thể download các file đã tìm kiếm.
* Hệ thống hỗ trợ nhiều định dạng file: txt, img…

1. **Searchable encryption**

* Searchable encryption giúp cho người dùng vừa đảm bảo dữ liệu được mã hoá, vừa giúp cho người dùng có thể tìm kiếm tài liệu dựa trên từ khoá. Vì quá trình này được thực hiện ở server (nơi lưu trữ dữ liệu) nên tất cả đều được thực hiện trên các file mã hoá để tăng tính bảo mật.
* Phân loại searchable encryption: gồm 2 loại là: SSE (symmetric searchable encryption) và PEKS (public encryption with keyword search). Trong đề tài này, chúng em sử dụng SSE cho việc mã hoá dữ liệu.
* SSE: hoạt động dựa trên 1 khoá bí mật của người dùng (private key) để mã hoá tài liệu và tạo trapdoor cho việc tìm kiếm.

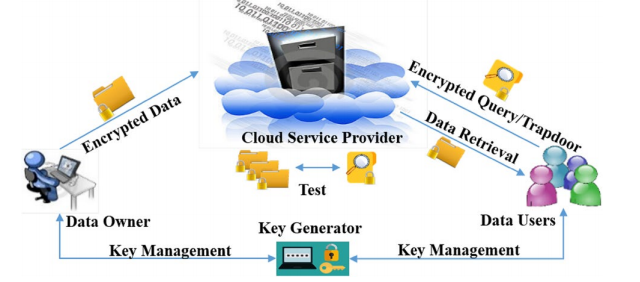


Figure 2 Mô hình SSE (trích trong bài báo tham khảo (3))

* Các thành phần của lược đồ gồm có 4 thực thể: data owner (người muốn gửi tài liệu), data user (người dùng hợp pháp muốn sử dụng tài liệu), cloud service provider, key generator.
* Có hai loại SSE, tĩnh và động. SSE tĩnh là nơi dữ liệu được chuẩn bị và tải lên máy chủ lưu trữ một lần và sau đó chỉ có các truy vấn tìm kiếm được thực hiện. SSE động cho phép thêm, xóa hoặc sửa đổi dữ liệu sau lần tải lên đầu tiên. Nói chung, một kịch bản SSE tĩnh bao gồm sáu thuật toán:
* Key Generation Algorithm: *Keygen (1 k)* được thực thi bởi client và để tạo khoá bí mật K.
* Keyword Index Generation Algorithm: *BuildIndex (K, D)* được thực thi bởi client. Tạo index cho keyword, input là khoá bí mật K và tài liệu D.
* Symmetric encryption algorithm: *Encryption (D, K)* chạy ở phía client. Input: tập tài liệu D và khoá K. Output là tập ciphertext C.
* Keyword trapdoor generation algorithm: *Trapdoor (K, w)* chạy ở phía client. Input: khoá K và query keyword *w*. Output: trapdoor *Tw*.
* Keyword search algorithm: *Search (I, Tw)* chạy ở phía server. Input là trapdoor *Tw*  và index keyword I. Output: tập tài liệu D.
* Symmetric decryption algorithm: *Decryption (C, K)* Input: tập ciphertext C và khoá K. Output là tập tài liệu D. Chạy ở client.

1. **Hệ thống mã hóa tìm kiếm dữ liệu đa máy chủ (MS-SDC) được đề xuất**
2. **Kế hoạch và giải pháp**

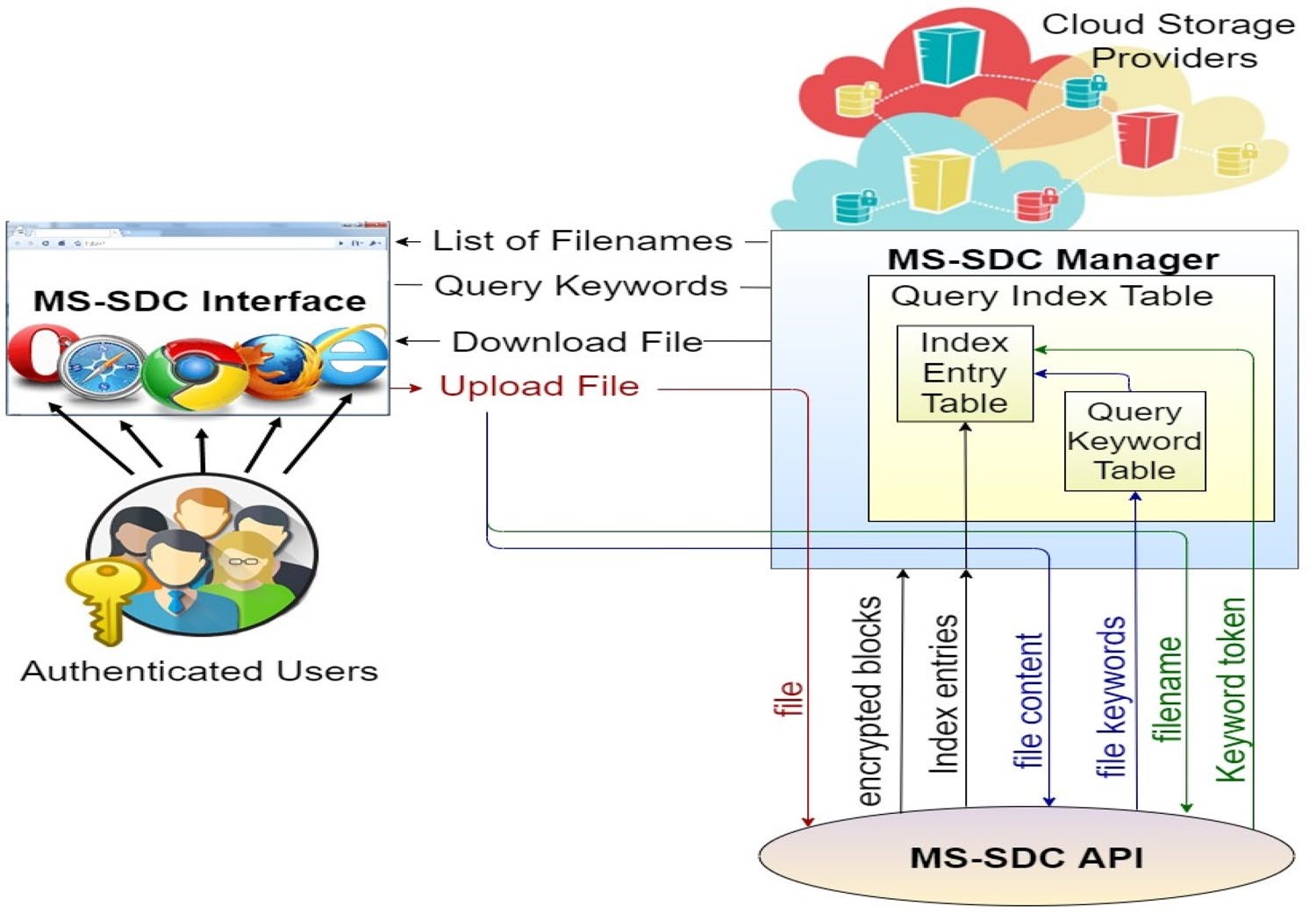
Hệ thống đề xuất được dành riêng để phân phối các tệp của người dùng vào nhiều máy chủ ở dạng mã hóa, người dùng được xác thực có khả năng tìm kiếm các tệp này. Kế thừa và mở rộng trong các nghiên cứu trước đó, bằng cách thêm các tính năng nâng cao, nó để tăng tính bảo mật và linh hoạt. Kiến trúc hệ thống có ba thành phần chính: MS-SDC Interface, MS-SDC Manager và MS-SDC API.

Figure 3 Sơ đồ MS-SDC (trích bài báo tham khảo (2))

1. **MS-SDC Interface**

Thành phần này chịu trách nhiệm cung cấp giao diện thích hợp cho hệ thống MS-SDC, cho phép người dùng tải lên và tải xuống các tệp, sau khi vượt qua quá trình xác thực. Hệ thống cung cấp cho người dùng cơ sở tìm kiếm các tệp bằng cách sử dụng các từ khóa trên các tệp được mã hóa, đã tải lên trước đó mà không cần phải giải mã chúng.

1. **MS-SDC Manager**

Thành phần này là thành phần quan trọng nhất trong hệ thống. Nó chịu trách nhiệm quản lý cơ sở dữ liệu máy chủ và cung cấp các chức năng cốt lõi của hệ thống:

1. Chuẩn bị tệp để tải lên và tải xuống.
2. Trích xuất từ khóa tài liệu cho mỗi tài liệu được tải lên.
3. Tạo và duy trì cả mục nhập chỉ mục và bảng từ khóa truy vấn.
4. Nối kết hai bảng trước đó để tạo thành bảng chỉ mục truy vấn.
5. Quản lý từ khóa truy vấn từ công cụ tìm kiếm để truy xuất các tài liệu tương ứng cho các từ khóa này được xếp hạng theo so sánh.
6. Băm dữ liệu trước khi được đưa vào cơ sở dữ liệu để có thêm mức độ bảo mật.

Các chức năng trên được thực hiện thông qua tương tác với MS-SDC API.

1. **MS-SDC API**

Thành phần này chịu trách nhiệm thực hiện sơ đồ MS-SDC và các thuật toán cần thiết.

* 1. *Chức năng tải lên*
  + Chủ sở hữu dữ liệu có thể tải lên các tệp mà họ muốn lưu trữ trên đám mây. Chủ sở hữu dữ liệu gửi tệp thông qua Giao diện MS-SDC, chuyển nó cho Trình quản lý MS-SDC. Chủ sở hữu dữ liệu có thể chọn số lượng máy chủ để lưu trữ tệp, nếu không Trình quản lý MS-SDC sẽ xác định số theo giao thức được sử dụng bởi Trình quản lý MS-SDC. Quản lý MS-SDC gọi MS-SDC API thích hợp để thực hiện chương trình SSE phân tán cơ bản.
  + MS-SDC API thực hiện các hành động để hoàn thành chức năng tải tệp lên, người dùng thêm tệp (document, image, video…) với các từ khóa được trích xuất dưới dạng từ khóa tệp, hệ thống cũng xem filename là một trong những từ khóa tệp.
  + Chức năng này chịu trách nhiệm chia tệp thành các khối được lập chỉ mục, chọn một máy chủ cho mỗi khối một cách ngẫu nhiên, mã hóa từng khối bằng khóa bí mật để cuối cùng có được một nhóm các khối được mã hóa, được phân phối đến nhiều máy chủ khác nhau. Nó cũng chịu trách nhiệm duy trì bảng nhập chỉ mục (Index Entry Table) và bảng từ khóa truy vấn (Query Keyword Table) cho các chức năng tiếp theo.
  1. *Chức năng tìm kiếm*

Cung cấp cho người dùng các chức năng tìm kiếm được thực hiện như sau:

1. Người dùng gửi một hoặc nhiều từ khóa thông qua MS-SDC Interface (trên công cụ tìm kiếm) lần lượt chuyển tiếp các từ khóa truy vấn này đến MS-SDC Manager.
2. MS-SDC Manager tìm kiếm bảng chỉ mục truy vấn để có được danh sách tên tệp khớp với các từ khóa đó (được xếp hạng theo mức độ phù hợp) với hỗ trợ MS-SDC API
3. MS-SDC Interface trình bày danh sách tên tệp cho người dùng.
   1. *Chức năng tải xuống*

Để download tệp từ các máy chủ lưu trữ khác nhau, trình quản lý MS-SDC thực hiện các bước sau:

1. Tìm kiếm trên bảng chỉ mục truy vấn bằng cách sử dụng FileName để lấy Token và BlockID.
2. Thu thập các khối được phân phối trên các máy chủ bằng cách sử dụng Token và BlockID để cuối cùng có được mỗi Block và Server Key của nó.
3. Giải mã từng Block theo Server Key của nó.
4. Phối/Sắp xếp các khối theo Block Index để xây dựng tệp gốc.
5. **Chi tiết các thuật toán của hệ thống được đề xuất**

Việc triển khai hệ thống được đề xuất đã được chia thành bốn thuật toán cụ thể là; FileUpload\_Setup, KeywordExtraction, FileDownload và SearchEngine\_ Keyword.

* 1. *FileUpload\_Setup*
* Thuật toán này sẽ hỗ trợ người dùng chia nhỏ file cần tải lên server thành các block nhỏ, mã hoá và gửi ngẫu nhiên đến các server.
* Thuật toán được thực hiện ở phía CLIENT.
* Các bước thực hiện thuật toán FileUpload\_Setup:

**Input**: File F, filename (keyword token), tập servers S = {S1, S2…., Sn}

**Output**: Tập các khối mã hóa C = {c1, c2, c3…, cx}, bảng I (Index Entry Table), tập các secret keys K.

*\* Khởi tạo:*

- Tạo danh sách trống t để lưu BlockID, các Block đã mã hoá (lưu id khối mã hoá và serverID giữ khối này)

- Khai báo danh sách các khối mã hoá C = {c1, c2, …., cx}

1. Tạo một Master Key duy nhất mk ¬ Pseudorandom function

Tạo keyword token: tok = HMACSHA3\_256(mk, filename)

1. Keygen Algorithm: Tạo danh sách secret keys cho các server:

K = HMACSHA3\_256(mk, (k1,k2,k3,…, kn))

1. Chia F thành các khối F = {f 1, f2,…, fx}

Xét qua tất cả các khối của F là f

1. Chọn ngẫu nhiên một server: i = random(1..n)
2. e ¬ Encryption AES256-CTR(Ki , IVi , f)
3. Tạo Block IDs: id(e)= HMACSHA3\_256(tok, e)

* Lưu e và id(e) vào cj (trong đó j = id(e), cj= e)
* Nối e­và j vào t

1. Lưu block lên Server (Si)
2. Đặt I[tok] = t

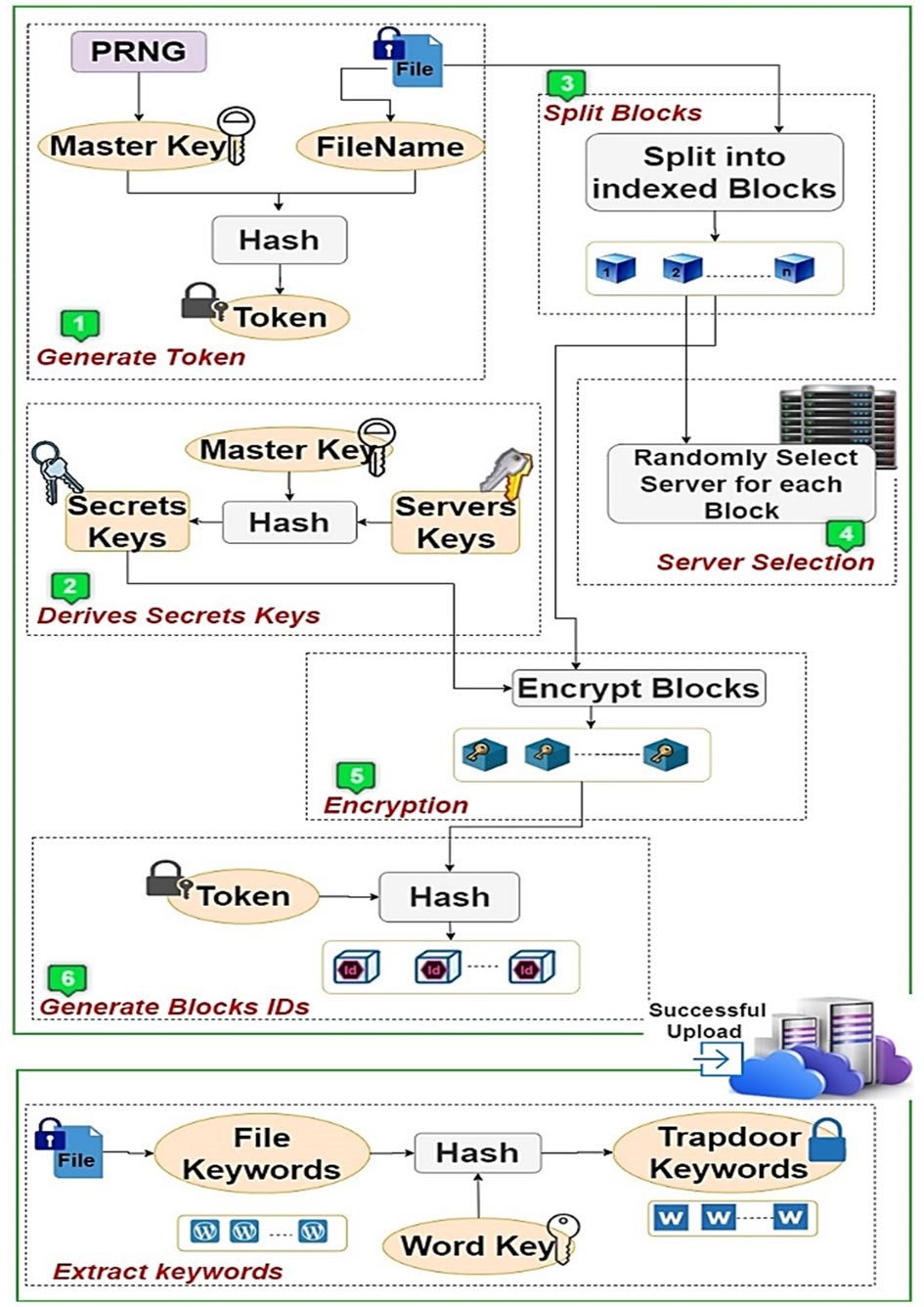


Figure 4 Sơ đồ thuật toán Upload và KeyExtraction (trích bài báo tham khảo (2))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| FileName | Token | Block Id | Block Index | Server Name | Server Key |
| Algorithm | Asdfg463 | U34n2n2l | 1 | S2 | Gsgi3nds |
| Algorithm | Asdfg463 | Uaet4354 | 2 | S1 | Ukngoek2 |
| Introduction | Qweg564i | Rfv3454n | 6 | S1 | Ukngoek2 |

Bảng 1 Ví dụ về Index Entry Table

* 1. *KeywordExtraction*
* Thuật toán này sẽ tạo bảng query keyword table được dùng cho quá trình tìm kiếm tài liệu sau này.
* Khi người gửi tài liệu sẽ gửi kèm file keyword.
* Thuật toán này sẽ được thực hiện ở CLIENT.
* Các bước thực hiện:

**Input**: file keyword Fw = {Fw 1 , …., Fwy}, index entry table I, word key wk, token của file: tok

**Output**: Querry Keyword Table W

1. Xây dựng trapdoor (hàm này được sử dụng sau cho việc tìm kiếm):

Tw i ¬ HMAC-SHA3-256(wk, Fw i ) (i chạy từ 1 đến y)

1. Thêm Twi||tok vào bảng W. (i chạy từ 1 đến y)

|  |  |
| --- | --- |
| Token | Keywords |
| Asdfg463 | Gsgi3nds |
| Asdfg463 | Mndive12 |
| Asdfg463 | Ukngoek2 |
| Qweg564i | Gsgi3nds |
| Qweg564i | Ukngoek2 |

Bảng 2 Ví dụ về Query Keyword Table

* 1. *SearchEngine\_ Keyword*
* Thuật toán này thực hiện chức năng tìm kiếm file cho người dùng và trả về các danh sách filename liên quan đến keyword mà người dùng nhập vào.

**Input:** Tập các query keyword Qw = {Qw 1 , …., Qwy}, word key wk, index entry table I.

**Output:** Danh sách các Filenames (F i )

1. Tạo trapdoor từ keywords sử dụng cùng hàm hash khi tạo trapdoor ở server: *Tw i = HMAC-SHA3-256(wk, Qw i ) (i chạy từ 1 đến y),* sau đó gửi Tw cho Server.
2. Thuật toán search: If Tw i  khớp với entry trong W (i chạy từ 1 đến y)

Nếu tìm thấy sẽ trả về danh sách các toki cho user.

Nếu không tìm thấy, sẽ không trả về gì.

1. Từ I[toki]: trích xuất Filename (Fi)

* Trong đó: ở bước (1) và (3) sẽ được thực hiện ở CLIENT, bước (2) sẽ được thực hiện trên Server.

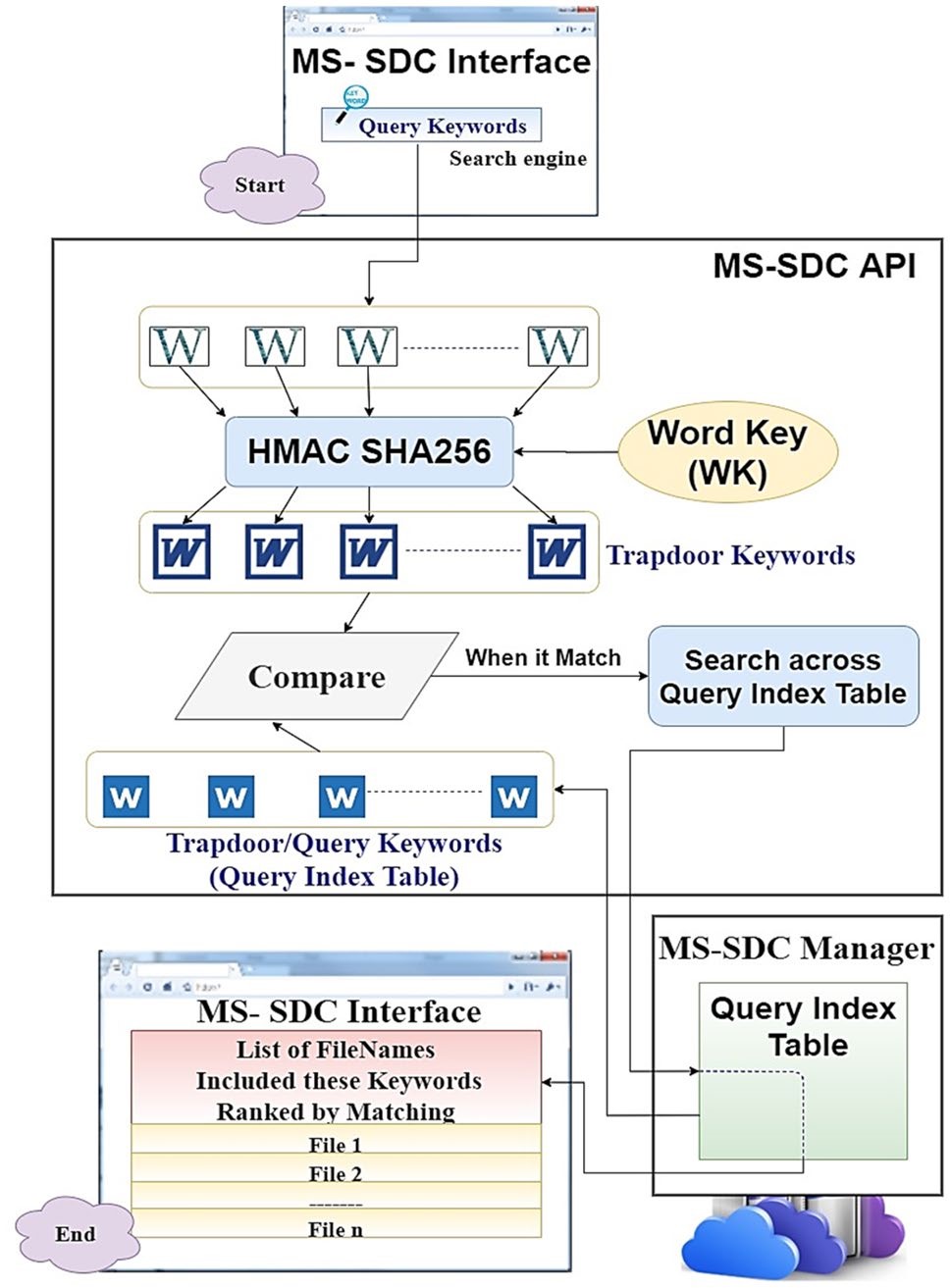


Figure 5 Sơ đồ chi tiết thuật toán Search (trích bài báo (2))

* 1. *FileDownload*
* Hàm này có chức năng sẽ download file về sau khi người dùng đã tìm kiếm được file mình cần.
* Để có thể download file, trước hết cần phải thu thập lại các block đã phân tán ở các server, giải mã và nối các block lại.

**Input:** Filename(F), masterkey mk

**Output:** File output (Fo)

1. tok = HMAC-SHA3-256(mk, F)
2. If toknot in I

Return nothing, break;

Else

Từ I[tok]: lấy BlockID, serverID, Server key K, indexServer từ đó lấy các khối mã hoá về.

1. Decrypt mỗi khối -> Decryption(K j , f i ) (j là index của server)
2. Từ Block index: Nối các khối fi đã giải mã lại thành file output hoàn chỉnh (Fo) và sẵn sàng cho việc download.

* Các bước trên đều được thực hiện ở CLIENT.

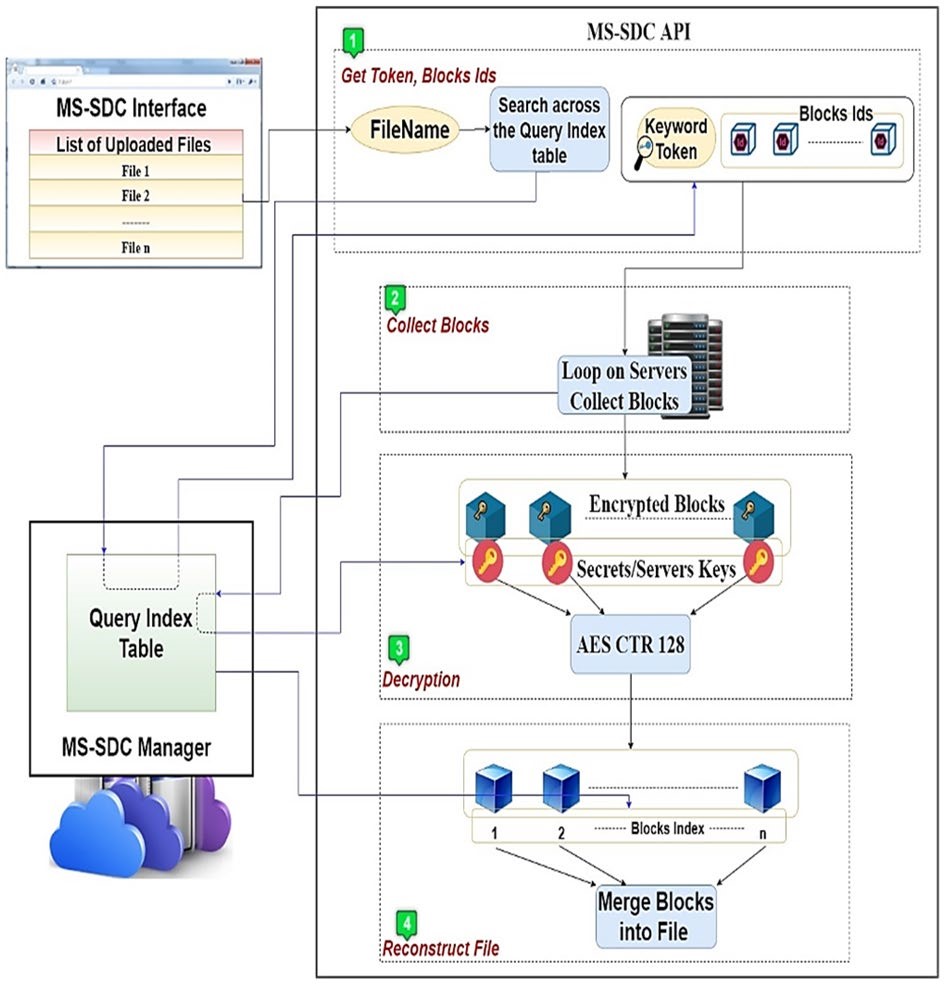


Figure 6 Sơ đồ chi tiết thuật toán Download (trích bài báo tham khảo (2))

1. **Contributions**

* Cho phép các tài liệu được mã hóa theo khối, lưu trữ ở các clouds khác nhau, đảm bảo với xác suất cao rằng không có một bên thứ 3 duy nhất nào có được một bộ khối hoàn chỉnh, sử dụng để tìm hiểu thông tin về tài liệu.
* Giảm thiểu thời gian tải lên và thời gian tìm kiếm so với những sản phẩm được nghiên cứu trước đó. Đồng thời khắc phục một số hạn chế của những nguyên cứu trước đó, cải tiến thêm một vài tính năng mới: Cho phép tạo khóa chính ngẫu nhiên cho mỗi tệp đã tải lên, kích thước, số lượng máy chủ được cấu hình, không còn cố định như trước, áp dụng cho hình ảnh, video và các loại khác,…

1. **Comments**

* Ưu điểm:
* Mô hình MS-SDC sử dụng khóa ngẫu nhiên cho các tệp tải lên. Giả sử, khóa chính được sử dụng cho tất cả các tệp được tải lên thì sẽ tăng nguy cơ lộ khóa chính. Khi attacker có quyền kiểm soát tất cả các máy chủ lưu trữ thì lúc này attacker sẽ biết được thông tin rò rỉ từ tất cả các máy chủ lưu trữ liên quan đến hệ thống.
* Hệ thống đã sử dụng từ khóa truy vấn ở dạng băm. Nếu như sử dụng từ khóa gốc (dạng plaintext), thì đối với thuật toán Search và Keyword Extraction, attacker có thể biết được block đã được mã hóa được liên kết với keyword token trong quá trình tải xuống tệp. Bên cạnh đó, attacker có thể tìm hiểu nội dung của tài liệu bằng cách lặp lại các từ khóa truy vấn được sử dụng để tìm kiếm ở phía máy chủ.
* Hạn chế:
* Hệ thống MS-SDC chỉ hoạt động tĩnh, người dùng chỉ có thể upload file, search, và download file. Người dùng còn không có khả năng update và delete tài liệu trên hệ thống.
* Mặc dù hệ thống hiện tại đã đảm bảo với xác suất cao cho dữ liệu người dùng không bị rò rỉ, lộ dữ liệu, vì tất cả đều đã được mã hoá và phân tán tới các server một cách an toàn. Tuy nhiên hệ thống vẫn còn nhiều hạn chế: hệ thống chỉ làm việc với file, với nhu cầu xã hội ngày nay, hệ thống thường sẽ được dùng bởi những tổ chức, cơ quan, doanh nghiệp. Và họ thường làm việc nhiều với database cho việc quản lí nhân sự, hàng hoá,….Chính vì vậy một hướng phát triển cho hệ thống đó là hỗ trợ thêm cả phần DBMS (cả SQL lẫn no SQL).

1. **Demo**
2. **Triển khai:**

* Thư viện sử dụng: Vector để lưu thông tin của các tables, CryptoPP sử dụng để mã hóa và giải mã thông tin.
* Sử dụng hàm HMAC-SHA3-256 làm hàm hash.
* Sử dụng hàm AES-256-CTR để mã hoá và giải mã các blocks.
* Kích thước block đặt mặc định là 4KB (mặc dù trong bài báo kích thước của các block là tùy biến).
* Số lượng server quy ước cố định là 5 servers.
* Cấu hình máy demo: Môi trường triển khai chạy trên CPU Intel(R) Core(TM) i7-8565U@ 1.80GHz - 1.99 GHz với 8 GB bộ nhớ RAM.

1. **Demo**

* Link video demo: [Video\_Demo\_FinalProject\_Cryptography.wmv](https://drive.google.com/file/d/1pZ0zFb4VveiQAZAT7vfyBZB0mF_I6s0G/view?usp=sharing)
* Các câu lệnh mà chương trình demo hỗ trợ:
  + Uploadfile: filename là tên file của file cần upload, filekeyWord là tên file text chứa danh sách các key word
  + Search: keyword là một từ khóa
  + Downloadfile: từ danh sách đã tìm kiếm
  + Ngoài ra phần demo còn cho phép kiểm tra thông tin của các bảng: [QueryKeywordTable](#QueryKeywordTable), [ServerList](#ServerList), [IndexEntryTable](#IndexEntryTable).

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 7 Giao diện chương trình khi gõ lệnh --help

* ***Tính năng Upload file lên Server***
  + Uploadfile “headline.json” với danh sách các keyword được lưu ở file “headline”.
  + Các từ khóa bao thêm vào bao gồm: headline, ML, Deadline, dataset, ngoài ra filename của tệp cũng là 1 từ khóa (headline.json).
  + Mô tả quả quá trình hoạt động: sau khi người dùng sử dụng lệnh *uploadfile headline.json headline*, trong đó file *headline.json* là file chứa nội dung, còn file *headline* chứa từ khoá cho việc tìm kiếm. Tất cả các quá trình sau đây đều sẽ được thực hiện ở phía client. Bước đầu tiên để thực hiện thuật toán này là tạo token: chương trình sẽ tạo ngẫu nhiên khoá được gọi là Master Key. Sau đó sẽ sử dụng HMAC-SHA3-256 Master Key vừa được tạo với FileName để tạo ra token. Bước thứ 2 là tạo các Secrets Key phục vụ cho việc mã hoá các khối bằng cách HMAC-SHA3-256 MasterKey với Server Keys (mỗi server sẽ có 1 secret key khác nhau). Bước thứ 3 chính là chia file *headline.json* thành các block nhỏ, kích thước sẽ do người dùng quy định (trong demo này mỗi block sẽ có kích thước: 4KB). Như vậy file *headline.json* sẽ được chia làm 5 blocks (như trong [figure 9](#Figure8)). Bước thứ 4 chính là chọn ngẫu nhiên server sẽ lưu trữ các blocks. Sau khi thực hiện các bước trên xong thì bước thứ 5 chính là mã hoá các khối sử dụng thuật toán AES256-CTR với secret keys ứng với server đã chọn. Và bước cuối cùng, bước thứ 6 chính là tạo BlockID (BlockID nằm ở sau Server trong figure 9) bằng cách HMAC-SHA3-256 token với các block đã mã hoá. Tất cả các thông tin: filename, token, block index, Blockade, Server Name….. đều được lưu trữ trong bảng [Index Entry Table](#IndexEntryTable) ở client. Và như vậy các khối đã sẵn sàng cho việc upload lên server. Song song với việc uploadfile thì chương trình cũng đồng thời tạo [query keyword table](#QueryKeywordTable) (trapdoor) cho việc tìm kiếm sau này. Đầu tiên chương trình sẽ trích xuất các keyword trong file *headline* (ở đây các keyword là: headline.json, headline, ML, Deadline, dataset như trong [figure 9](#Figure8)) và HMAC-SHA3-256 với Word Key để tạo ra trapdoor keyword (chính là cột keyword trong [Query Keyword Table](#QueryKeywordTable)) tiếp theo sẽ tạo bảng tương ứng với các keyword với token. Sau đó sẽ gửi bảng này cho SERVER lưu trữ phục vụ cho mục đích tìm kiếm sau này.
  + File được upload lên được chia làm 5 blocks, thời gian mã hóa và lưu vào các thư mục server là 50 milliseconds

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 8 Giao diện chương trình khi thực hiện uploadfile

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 9 Query Keyword Table

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 10 Index Entry Table

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 11 Server List

* ***Tính năng tìm kiếm dựa trên từ khóa***
  + Tìm kiếm từ khóa ML.
  + Mô tả quá trình hoạt động: Với từ khoá *ML*, ở phía client sẽ tạo trapdoor keyword bằng cách HMAC-SHA3-256 *ML* với Word Key, sau đó sẽ gửi trapdoor keyword này lên cho SERVER tiến hành tìm kiếm. Tại server, sau khi nhận được trapdoor keyword, server sẽ so sánh trapdoor keyword với cột keyword ứng với bảng [Query Keyword Table](#QueryKeywordTable), nếu như trùng khớp hàng nào trong bảng, SERVER sẽ trả về token tương ứng (với từ khoá *ML*, server sẽ trả về token ứng với 2 file *headline.json* và *id2t.csv*). Client sau khi nhận được token, sẽ tra bảng [Index Entry Table](#QueryKeywordTable) và hiển thị ra tên file tương ứng cho người dùng.
  + Kết quả trả về 2 file là “headline.json”, “id2t.csv”
  + Vì thông tin các bảng được lưu trữ trên [vector] của C++ do vậy thời gian tìm kiếm nhỏ hơn nhiều so với demo của bài báo… là 29 microseconds

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 12 Giao diện chương trình khi thực hiện Search

* ***Tính năng download file dựa trên tên file***
  + Download file headline.json (có index = 0 từ danh sách vừa tìm kiếm được)
  + Mô tả quá trình hoạt động: Để download được file *headline.json*, bước thứ nhất chương trình sẽ lấy token và các BlockIDs tương ứng với *headline.json* bằng cách tìm kiếm trong [Index Entry Table](#IndexEntryTable). Bước thứ 2, chúng ta sẽ thu thập lại các blocks (5 blocks ứng với file *headline.json*) phân tán trên các server dựa vào token và BlockIDs ở bước 1. Sau khi đã thu thập hết các blocks, bước thứ 3, chúng ta sẽ tiến hành giải mã các khối sử dụng SecretKeys (chính là cột ServerKey trong bảng [Index Entry Table](#IndexEntryTable)). Bước cuối cùng, chúng ta sẽ sắp xếp lại các block theo đúng thứ tự dựa vào Block Index và hợp lại thành file hoàn chỉnh cho người dùng có thể download.
  + Kết quả thu được lưu ở thư mục Download

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 13 Giao diện chương trình khi thực hiện Download

Ảnh có chứa văn bản

Mô tả được tạo tự động

Figure 14 File headline.json đã được download trong thư mục Download

# ***TÀI LIỆU THAM KHẢO***

* ***Book:***

1. *Pro Cryptography and Cryptanalysis with C++20*

* ***Paper:***

1. *Shahien, T., Sarhan, A. M., & Alshewimy, M. A. (2020). Multi-server searchable data crypt: searchable data encryption scheme for secure distributed cloud storage. Journal of Ambient Intelligence and Humanized Computing, 1-19.*
2. *Poh, Geong Sen, et al. "Searchable data vault: encrypted queries in secure distributed cloud storage." Algorithms 10.2 (2017): 52.*

# ***NHẬN XÉT CỦA GIẢNG VIÊN***

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………