

بازیابی فازی داده‌های رمزشده با استفاده از داده‌ساختارهای چندمنظوره

اعظم سلیمانیان^{۱*} و شهرام خزایی^۲

^۱دانشگاه خوارزمی تهران، دانشکده ریاضی و علوم کامپیوتر، تهران، ایران

^۲دانشگاه صنعتی شریف، دانشکده علوم ریاضی، تهران، ایران

چکیده

با گسترش روزافزون سرویس‌های ابری، افراد حقیقی و حقوقی بیشتری تمایل به برونو سپاری داده‌های خود روی این سرویس‌ها دارند؛ اما بهدلایل امنیتی ترجیح می‌دهند قبل از برونو سپاری داده آن را رمز کنند. رمزکردن داده به روش‌های معمول می‌تواند موجب ایجاد اختلال در عملکرد سرویس ابری، مانند عملکرد جستجو شود. روش‌های رمزگذاری جستجوپذیر به عنوان ابزاری مناسب، امکان جستجو روی داده رمزشده را فراهم می‌سازند. با توجه به نیازهای متنوع کاربران، توسعه عملکردهایی که این روش‌ها قادر به پشتیبانی آن‌ها هستند مورد توجه قرار گرفته است. یکی از این عملکردها جستجوی رتبه‌بندی شده است که نتایج را با توجه به میزان ارتباطی که با واژه مورد جستجو دارند، به صورت رتبه‌بندی شده در اختیار کاربر قرار می‌دهد؛ بنابراین تنها با ارسال اسناد مرتبه‌تر می‌توان ترافیک شبکه را کاهش داد. داده‌ساختارها به عنوان بلوک‌های سازنده در رمزگذاری جستجوپذیر متقاضان محسوب می‌شوند و تنوع در این داده‌ساختارها منجر به دست‌یابی به سطوح متنوع از امنیت، کارایی و عملکرد می‌شود. از سوی دیگر، برای رتبه‌بندی اسناد معیارهای متفاوتی وجود دارد. در این مقاله، معیار بازیابی فازی استناد در نظر گرفته شده است که با وجود کارایی بالا و سادگی، تا کنون در مبحث جستجو روی داده رمزشده به کار گرفته نشده است. برای این منظور، به بررسی داده‌ساختارهایی می‌پردازیم که امکان دستیابی به عملکرد جستجوی رتبه‌بندی شده را فراهم می‌سازند. ترکیب داده‌ساختار ارائه شده با معیار بازیابی فازی، روش جستجوی رتبه‌بندی شده‌ای را فراهم می‌آورد که علاوه بر کارایی، امنیت داده را نیز تضمین می‌کند.

وازگان کلیدی: رمزگذاری جستجوپذیر، جستجوی رتبه‌بندی شده، ساختمن داده، بازیابی فازی، پرسمان بولی.

Fuzzy retrieval of encrypted data by multi-purpose data-structures

Azam Soleimanian^{1*} & Shahram Khazaei²

¹Department of Mathematics and Computer Science, Kharazmi University, Tehran, Iran

²Department of Mathematical Science, Sharif University of Technology, Tehran, Iran

Abstract

The growing amount of information that has arisen from emerging technologies has caused organizations to face challenges in maintaining and managing their information. Expanding hardware, human resources, outsourcing data management, and maintenance an external organization in the form of cloud storage services, are two common approaches to overcome these challenges; The first approach costs of the organization is only a temporary solution. By contrast, the cloud storage services approach allows the organization to pay only a small fee for the space actually in use (rather than the total reserved capacity) and always has access to the data and management tools with the most up-to-date mechanisms available. Despite the benefits of cloud storage services, security challenges arise because the organization's data is stored and managed outside of the most important organization's supervision. One challenge is

* Corresponding author

*نویسنده عهده‌دار مکاتبات

confidentiality protection of outsourced data. Data encryption before outsourcing can overcome this challenge, but common encryption schemes may fail to support various functionalities in the cloud storage service. One of the most widely used functionalities in cloud storage services is secure keyword search on the encrypted documents collection. Searchable encryption schemes, enable users to securely search over encrypted data. Based on the users' needs, derivatives of this functionality have recently been considered by researchers. One of these derivatives is ranked search that allows the server to extract results based on their similarity to the searched keyword. This functionality reduces the communication overheads between the cloud server and the owner organization, as well as the response time for the search. In this paper, we focus on the ranked symmetric searchable encryption schemes. In this regard, we review structures proposed in the symmetric searchable encryption schemes, and show that these two data structures have capabilities beyond their original design goal. More precisely, we show that by making the data structures, it is possible to support secure ranked search efficiently. In addition, by small changes on these data, we present two ranked symmetric searchable encryption schemes for single keyword search and Boolean structures which introduced-keyword search based on the data.

Keywords: Searchable encryption, Ranked search, Linked list, Lookup table, Fuzzy retrieval, Boolean query.

داده‌ساختارهای مختلف سطوح متفاوتی از امنیت، کارایی و عملکرد را فراهم می‌سازند که بسته به نیاز، می‌توان از آن‌ها استفاده کرد.

از جمله عملکردهای موردتوجه در رمزگذاری جستجوپذیر، جستجوی رتبه‌بندی شده است. جستجوی رتبه‌بندی شده این امکان را برای سرور فراهم می‌سازد تا نتایج را با توجه به میزان ارتباط آن‌ها به واژه مورد جستجو، به صورت رتبه‌بندی شده در اختیار کاربر قرار دهد. این امر باعث کاهش بار ترافیکی شبکه و افزایش کارایی می‌شود، زیرا سرور می‌تواند مرتبط‌ترین اسناد را به کاربر بازگرداند که تعداد آن‌ها توسط کاربر تعیین می‌شود؛ همچنین این امکان نیز وجود دارد که اسنادی برگردانده شوند که میزان ارتباط آن‌ها به واژه مورد نظر، از یک پارامتر تعیین‌شده توسط کاربر بیشتر باشد.

حال باید مشخص کنیم منظور از میزان ارتباط سند به یک واژه خاص چیست و چگونه سنجیده می‌شود. به طور کلی میزان ارتباط سند به واژه را فراوانی نسبی آن واژه در سند تعریف می‌کنیم. ملاک‌های متفاوتی برای تعیین ارتباط سند به واژه موجود است: از جمله، معیار TF-IDF³ و معیار ارتباط فازی (بازیابی فازی). معیار TF-IDF که در رمزگذاری جستجوپذیر نیز مورد استفاده قرار گرفته است [22]، به دانش دقیق در مورد تکرار واژه در سند نیاز دارد (فرمول ریاضی این معیار را در [22] ببینید). متأسفانه، این معیار برای پرسمان‌های بولی (زمانی که داده‌ها رمزشده هستند) به راحتی قابلِ اعمال نیست. در بازیابی فازی، با وجود عدم قطعیت در مورد حضور واژه در سند، جستجو همچنان امکان‌پذیر خواهد بود و همچنین با به کارگیری قوانین نظریه فازی، امکان اعمال کارایی پرسمان‌های بولی⁴ فراهم می‌شود.

³ Term frequency-inverse document frequency

⁴ Boolean query

۱- مقدمه

با گسترش خدمات ابری، مسئله تأمین امنیت داده‌ها در ابر به یکی از چالش‌های بزرگ در این حوزه تبدیل شده است. روش‌های رمزگذاری معمول جهت حفظ محramانگی داده، منجر به ایجاد اختلال در برخی عملکردهای سرویس ابری، مانند عملکرد جستجو می‌شوند؛ ازین‌رو، روش‌های رمزگذاری خاص‌منظوره، هم‌چون رمزگذاری جستجوپذیر، پا به عرصه ظهور نهاده‌اند که هم‌زمان با حفظ محramانگی داده، عمل جستجوی روی داده را نیز ممکن می‌سازند.

رمزگذاری جستجوپذیر در قالب پروتکل زیر بین مالک داده، سرویس‌دهنده ابری و کاربران اجرا می‌شود. مالک داده مجموعه‌ای از اسناد در اختیار دارد؛ به طوری که هر سند شامل تعدادی واژه است. مالک یک شاخص¹ متناظر با اسناد خود تولید می‌کند که به طور کارا مشخص می‌کند هر واژه در چه اسنادی واقع شده، سپس شاخص و اسناد را بدون توجه به این‌که هر سند شامل چه واژه‌هایی است، رمز می‌کند؛ درنهایت، شاخص رمزشده و اسناد رمزشده را روی سرور ابر قرار می‌دهد. کاربری که تمایل به جستجو دارد، پرسمان خود را در قالب یک نشان² (که تابعی از یک کلید و واژه مورد نظر است) به سرور ابر ارسال می‌کند. سرور با استفاده از این نشان، جستجو را اعمال می‌کند و اسنادی که واژه مورد نظر را شامل می‌شوند به کاربر بر می‌گرداند؛ درنهایت کاربر نتایج را رمزگشایی می‌کند.

رمزکردن شاخص یادشده به روش‌های مختلفی امکان‌پذیر است. به طور کلی در همه روش‌ها شاخص در قالب یک داده‌ساختار رمز می‌شود و تفاوت این روش‌ها در نوع داده‌ساختاری است که مورد استفاده قرار گرفته است.

¹ Index

² Token

طرح ارائه شده در [8] از داده‌ساختاری استفاده می‌کند که مشکل از یک آرایه^۳، یک فهرست پیوندی^۴ و یک جدول مراجعه^۵ است. این داده‌ساختار جستجوی تکوازه را در زمان بهینه (یعنی از مرتبه تعداد استنادی که واژه را شامل می‌شوند)، فراهم می‌کند. آن‌ها همچنین داده‌ساختاری ارائه کردن که به قیمت افزایش تعداد نشان ارسالی از سمت کاربر، امنیت را افزایش می‌دهد. کش و همکارانش [6] داده‌ساختاری ارائه کردن که تنها با ارسال یک نشان، سطح امنیت بالایی را (در مدل اوراکل تصادفی) تضمین می‌کند. این روش همچنین قابلیت و کارایی قابل توجهی در پشتیبانی از پرسمان‌های متتنوع دارد.

به‌طور کلی، تمام روش‌های موجود در رمزگذاری جستجوپذیر، با طراحی مناسب داده‌ساختار، می‌توانند به هدف خود اعم از امنیت، کارایی یا عملکرد دست یابند. برای نمونه داده‌ساختارهای ارائه شده در [19, 14, 7, 9, 5, 3] برای پشتیبانی از جستجوهای پویاء، موارد معرفی شده در [11, 6] برای پشتیبانی از پرسمان‌های بولی و کارهای [23, 22, 21, 4] برای پشتیبانی از جستجوی رتبه‌بندی شده معرفی شده‌اند؛ همچنین داده‌ساختارهای مناسب برای پشتیبانی از جستجوی فازی و پرسمان‌های متتنوع روی پایگاه داده رابطه‌ای⁶ بهتری در [15, 2] و [12] یافت می‌شوند.*

در [21] یک روش جستجوی رتبه‌بندی شده برای تکوازه ارائه شده است. در این روش، سرور برای بازیابی شناسه استناد با بیشترین میزان ارتباط به واژه، ابتدا باید تمام شناسه استنادی که واژه را شامل می‌شوند، بازیابی و سپس مرتبطترین استناد را مشخص کند. در روش ما، سرور بدون نیاز به بازیابی تمام شناسه استناد شامل واژه، می‌تواند به‌طور مستقیم تنها استناد مرتبطتر را بازیابی کند که این خود باعث افزایش کاری می‌شود. در [23]، یک روش

³ Array⁴ Linked list⁵ Lookup table⁶ Dynamic

* در جستجوی پویاء، مالک داده مجاز است در هر زمان دلخواه، استنادی را از مجموعه حذف یا به مجموعه اضافه کند و یا متن استناد را تغییر دهد. در جستجوی فازی، تنها استناد شامل واژه تحت پرسمان، بلکه استناد شامل واژه‌های مشابه آن نیز بازیابی می‌شوند. کاربرد اصلی این روش زمانی است که کاربر دانش دقیقی از املای واژه ندارد یا واژه مورد جستجو حاوی غلط املایی است.

⁷ Relational database

نوآوری: در این مقاله، دو داده‌ساختار موجود در رمزگذاری جستجوپذیر را مرور می‌کنیم و نشان می‌دهیم این داده‌ساختارها قابلیت‌هایی بیش از هدف اصلی طراحی‌شان دارند. به‌طور دقیق‌تر، نشان می‌دهیم که با ایجاد تغییرات اندکی در این داده‌ساختارها، امکان پشتیبانی از جستجوی رتبه‌بندی شده به‌طور کارایی وجود دارد؛ اما نکته‌ای که باید مورد توجه قرار گیرد، کشف چنین قابلیتی در داده‌ساختارهای یادشده است؛ زیرا به سختی می‌توان داده‌ساختارهایی را پیدا کرد که اعمال تغییرات کوچک در آن‌ها، منجر به پشتیبانی کارا از عملکرد جدیدی شود.

با توجه ویژه به چنین قابلیتی در داده‌ساختارهای مورد نظر و اعمال تغییرات ساده در آن‌ها، نوآوری‌های زیر را خواهیم داشت:

- بازیابی فازی را به عنوان روشی کارا، برای نخستین بار در جستجوی روی داده‌های رمزشده به کار می‌بریم.
- طرح رمز جستجوی رتبه‌بندی شده‌ای ارائه خواهیم کرد که قادر است در زمان بهینه، با دریافت پارامتر α از سمت کاربر، کلیه استنادی را که میزان ارتباط آن‌ها با واژه مورد جستجو دست کم به میزان α است، شناسایی کند.
- طرح رمز جستجوی رتبه‌بندی شده‌ای ارائه می‌کنیم که قادر است در زمان بهینه، مرتبطترین استناد متناظر با یک پرسمان بولی را بازیابی کند.

۲- کارهای مرتبه

نخستین طرح رمز جستجوپذیر در سال ۲۰۰۱ توسط Song و همکارانش [18] ارائه شد که یک طرح پویشی^۱ بود به این معنی که برای جستجوی واژه در یک سند، باید کل سند را بررسی می‌کرد. نخستین طرح رمز جستجوپذیر بر مبنای شاخص^۲، توسط گاه در سال ۲۰۰۳ ارائه شد [10] که توانست با حذف نیاز به پویش کل سند، کارایی را افزایش دهد. نخستین تعریف امنیت دقیق و مناسب با رمزگذاری جستجوپذیر، در سال ۲۰۰۶ توسط کورتمولاو همکارانش [8] ارائه و طرح‌هایی که امنیت مورد نظر را به صورت قابل اثبات برآورده می‌کردند، نیز ارائه شد.

با توجه به اهمیت شاخص در رمزگذاری جستجوپذیر، ارائه روش‌های متعددی برای رمزگذرن شاخص و ساخت داده‌ساختارها از روی شاخص مورد توجه قرار گرفت، که در ادامه به مهم‌ترین آن‌ها اشاره می‌شود.

¹ Thorough scan² Index-based

ذخیرهسازی دادهها در سمت سرور، به دست می‌آوریم؛ همچنین، در طرح خود، از یک معیار بازیابی فازی بهره می‌بریم که از جمله معیارهای استاندارد در بازیابی اطلاعات (روی داده رمزنشده) است. معیار استفاده شده، با توجه به ویژگی‌های ذاتی اش، به نوع خود نیز موجب ایجاد انعطاف در طراحی داده‌ساختارها لازم و متعاقباً بهبود کارایی می‌شود. ما در این کار از داده‌ساختارهای مطرح شده در [8] و [6] به ترتیب به منظور افزایش کارایی جستجوی رتبه‌بندی شده و پشتیبانی از پرسمان‌های متنوع استفاده می‌کنیم.

بنابراین، با ترکیب این دو داده ساختار و معیار بازیابی فازی که معیاری استاندارد، ساده و کارا است- یک روش جستجوی رتبه‌بندی شده به دست خواهیم آورد که در عین کارایی بالا پرسمان‌های متنوعی را نیز پشتیبانی می‌کند.

ساختار مقاله: در این مقاله، ابتدا در بخش ۳ به معرفی نمادها و ابزارهای لازم می‌پردازیم. از جمله، معیار بازیابی فازی و قوانین مربوطه در این بخش مرور خواهد شد. در بخش ۴، شیمای^۴ طرح رمز جستجوی رتبه‌بندی شده و مدل امنیت آن را ارائه خواهیم کرد. در بخش ۵، به بررسی داده‌ساختاری که کوتمولو و همکارانش [8] برای جستجوی تکوازه ارائه کرده‌اند، می‌پردازیم و نشان می‌دهیم که این داده‌ساختار قابلیت ترکیب با معیار بازیابی فازی و پشتیبانی از جستجوی رتبه‌بندی شده را دارد. در بخش ۶ داده‌ساختار ارائه شده توسط کش و همکارانش [6] را بررسی می‌کنیم و نشان می‌دهیم که چگونه تجهیز آن با معیار بازیابی فازی، امکان پشتیبانی از پرسمان‌های متنوع در جستجوی رتبه‌بندی شده را فراهم می‌سازد. در پایان، به نتیجه‌گیری و مقایسه روش‌های خود با کارهای مشابه می‌پردازیم.

۳- نمادگذاری

مجموعه اعداد طبیعی و صحیح را به ترتیب با \mathbb{N} و \mathbb{Z} نشان می‌دهیم. \mathbb{Z}_q نشان‌دهنده مجموعه اعداد صحیح به پیمانه عدد نخست^۵ است؛ یعنی، $\{1, 0, \dots, q\} = \mathbb{Z}_q$ که یک گروه جمعی به پیمانه q است. مجموعه رشته‌های n -بیتی را با $\{0, 1\}^n$ و مجموعه رشته‌های دودویی با طول متناهی را با^۶ $\{0, 1\}^\infty$ نشان می‌دهیم. نماد $y||x$ بیان‌گر عملگر پیوست دو رشته x و y است. نماد λ را برای نمایش پارامتر امنیتی به کار می‌بریم. منظور از یک الگوریتم P.P.T، یک الگوریتم احتمالی با زمان اجرای چندجمله‌ای، بر حسب λ است. همچنین $\text{poly}(\lambda)$ تابع چندجمله‌ای از λ را نمایش

⁴ Syntax

جستجوی رتبه‌بندی شده (با حضور چندین مالک داده) ارائه شده که تنها برای جستجوی تکوازه قابلِ اعمال است و از طرفی به بازیابی کل استناد نیاز دارد. در این طرح همچنین از نگاشت دو خطی^۱ (جورساز^۲) استفاده می‌شود که بهشت داده رتبه‌بندی شده چندوازه‌ای (رابطه فصلی چندوازه) ارائه شده که امنیت آن مطابق تعاریف رایج، اثبات نشده است؛ از طرفی معیار رتبه‌بندی به کار برده شده، معیار استاندارد در مبحث بازیابی اطلاعات^۳ نیست. این ایده در [22, 24, 22] از لحاظ کارایی بهبود داده شد و به علاوه از معیار استاندارد TF×IDF (که نیاز به دانش دقیقی در مورد تکرار واژه در سند دارد)، به جای یک معیار ابتکاری استفاده شد. با وجود بهبود حاصل شده، مدت زمان جستجو همچنان به تعداد استناد وابسته است. همچنین در [۱]، روشی مبتنی بر رمزگذاری نیمه‌هریخت و معیار TF×IDF برای رابطه فصلی واژه‌ها ارائه شد که علاوه‌بر سرور یک پردازنده نیز در سیستم حضور دارد به طوری که فرض عدم تبادی سرور و پردازنده مطرح است. ابتدا با استفاده از رمزگذاری نیمه‌هریخت، سرور میزان ارتباط استناد به پرسمان را به صورت رمزشده به دست می‌آورد؛ سپس مقادیر حاصل را به پردازنده ارسال و پردازنده با دسترسی به کلید رمزگشایی مقادیر را رمزگشایی و سپس دوباره رمزگذاری حافظه‌ترتیب را روی آن‌ها اعمال می‌کند و به سرور بر می‌گرداند. مشکل اساسی روش مطرح شده میزان پیچیدگی تعاملاتی بین سرور و پردازنده است که از مرتبه اندازه مجموعه استنادی است که پرسمان فصلی را برآورده می‌کند؛ به علاوه، پردازنده باید بهزای هر پرسمان عمل رمزگشایی مقادیر دریافتی از سرور و سپس عمل رمزگذاری حافظه‌ترتیب آن‌ها را انجام دهد؛ همچنین در [20]، از رمزگذاری تمام‌هریخت استفاده شده است که این گونه روش‌های رمزگذاری اغلب ناکارا هستند. در [16]، با استفاده از روش تسهیم راز به جای رمزگذاری نیمه‌هریخت، نیاز به رمزگذاری حافظه‌ترتیب از بین می‌رود؛ اما همچنان نیاز به بازیابی کل شناسه استنادی که پرسمان را برآورده می‌کند، وجود دارد.

روش‌هایی که ما ارائه می‌دهیم کاراتر، دارای امنیت قابل اثبات بر مبنای تعارف رایج امنیت و قابل اعمال روی پرسمان‌های متنوع‌تر هستند. به طور نسبی می‌توان گفت بهبودهای یادشده را به قیمت افزایش فضای لازم برای

¹ Bilinear map

² Paring

³ Information retrieval

$$id \in ID, \mu_F(id, w) \geq \alpha\}$$

که مجموعه‌اسناد با درجه عضویت کمینه α نسبت به واژه w را نمایش می‌دهد. چنانچه مجموعه بالا بر اساس مقادیر $(\dots, \mu_F(\dots, \mu_F(id, w), w), w)$ به ترتیب نزولی مرتب شود، مجموعه حاصل را با $DB_F(w, \alpha)$ نشان می‌دهیم.
همچنین، تعریف می‌کنیم:

$$DB_F(w) = \{\langle id, \mu_F(id, w) \rangle \mid id \in ID, \mu_F(id, w) > 0\} \quad (2)$$

و به شباهت $Ord(DB_F(w))$ بیان‌گر مرتب شده مجموعه $DB_F(w)$ است.

نمادهای روابط عطفی^۵، فصلی^۶ و نقیض^۷ را به ترتیب با \wedge ، \vee و \sim نشان می‌دهیم. قوانین زیر برگرفته از نظریه فازی مجموعه‌ها [17] هستند:

$$\begin{aligned} \mu_F(id, w_1 \wedge w_2) &= \min\{\mu_F(id, w_1), \mu_F(id, w_2)\}, \\ \mu_F(id, w_1 \vee w_2) &= \max\{\mu_F(id, w_1), \mu_F(id, w_2)\}, \\ \mu_F(id, \sim w) &= 1 - \mu_F(id, w). \end{aligned} \quad (3)$$

با استفاده از قوانین بالا برای پرسمان دلخواه q ، مجموعه $\mu_F(id, q)$ و بدنبال آن $Ord(DB_F(q, \alpha))$ قابل تعریف هستند؛ به عنوان مثال برای پرسمان:

$$(w_1 \wedge w_2) \vee w_3 = q \text{ داریم:}$$

$M_F(id, q) = \max \{ \min\{\mu_F(id, w_1), \mu_F(id, w_2)\}, \mu_F(id, w_3) \}$.
رمزگذاری حافظه‌تریب: نوعی رمزگذاری خاص منظوره برای رمزکردن داده‌های عددی است، بهنحوی که مقایسه آن‌ها و تعیین رابطه ترتیب را ممکن می‌سازد.
فرض می‌کنیم تابع رمزگذاری به صورت:

$Enc(K, .) : \mathcal{D} \rightarrow \mathcal{R}$
باشد که برای $M, N \in \mathbb{N}$ ، دامنه و برد آن به ترتیب زیر است:

$$\begin{aligned} \mathcal{D} &\stackrel{\Delta}{=} \{1, \dots, M\} \\ \mathcal{R} &\stackrel{\Delta}{=} \{1, \dots, N\} \end{aligned}$$

تعریف ۱ (رمزگذاری حافظه‌تریب). رمز متقاضی یقینی $\mathcal{OPF} = (Gen, Enc, Dec)$ را حافظه‌تریب گوییم اگر برای هر $[M] \ni i, j \in [M]$ و کلید K که توسط الگوریتم Gen تولید شده است، داشته باشیم:

$i < j \Leftrightarrow Enc(K, i) < Enc(K, j)$
امنیت رمزگذاری حافظه‌تریب به این معنی است که یک مهاجم کارا با دسترسی اوراکلی^۸ به تابع رمزگذاری و

می‌دهد. در این مقاله مدل مهاجم را صادق‌اما-کنجکاو (HBC^۱) در نظر می‌گیریم. به این معنی که مهاجم الگوریتم‌ها را به درستی دنبال، اما سعی می‌کند مانند یک شنودگر، از طریق مشاهده داده‌های رمزشده، پرسمان‌های دریافتی و جواب‌های متناظر شان، اطلاعاتی را به دست آورد. یک تابع ناچیز^۲ از λ با نماد $\text{negl}(\lambda)$ نشان داده می‌شود که از مرتبه $\lambda^{-\omega^{(1)}}$ است. به عبارت دیگر، $\text{negl}(\lambda)$ یک تابع ناچیز است اگر از وارون هر چند جمله‌ای سریع‌تر به سمت صفر حرکت کند. دسترسی اوراکلی مهاجم \mathcal{A} به الگوریتم B را با نماد $\mathcal{A}^{B(.)}$ نمایش می‌دهیم و به این معنی است که مهاجم بدون درگیرشدن در نوع محاسبات تنها از طریق پرسش و پاسخ به الگوریتم دسترسی دارد. فرض می‌کنیم مجموعه اسناد به شکل زیر باشد:

$DB = (W_1, W_2, \dots, W_d)$
به طوری که هر سند i مجموعه‌ای از چندین واژه و با $id_i \in \{0,1\}^n$ و $W_i \subseteq \{0,1\}^*$ باشد که $W = \bigcup_{i=1}^d W_i$ بیان‌گر همه واژه‌های متمایز در مجموعه اسناد است که آن را واژه‌نامه می‌نامیم. پارامترهای d و n چند جمله‌ای از λ هستند. جهت سهولت، شناسه اسناد را اعداد طبیعی در نظر می‌گیریم، بنابراین، مجموعه:
 $ID = \{id \mid W_{id} \in DB\} = \{1, 2, \dots, d\}$
بیان‌گر مجموعه همه شناسه‌ها است. همچنین، $DB(w)$ نشان‌دهنده شناسه اسنادی است که واژه w را شامل می‌شوند؛ یعنی، $DB(w) = \{id \mid w \in W_{id}\}$. الگوی دسترسی^۳ به شناسه اسنادی گفته می‌شود که پرسمان را برآورده می‌کنند (به عنوان مثال برای تکوازه w الگوی دسترسی $DB(w)$ است) و الگوی پرسمان^۴ تکراری بودن یا نبودن پرسمان را مشخص می‌کند. برای مشاهده تعریف دقیق الگوی دسترسی و الگوی پرسمان، خواننده به [8] ارجاع داده می‌شود.

توصیف فازی مجموعه اسناد: مجموعه زیر توصیف فازی مجموعه اسناد را نشان می‌دهد:

$$\begin{aligned} F &= \{\langle id, w, \mu_F(id, w) \rangle \mid id \in ID, w \in W\} \\ \text{که } \mu_F : ID \times W \rightarrow [0,1] \text{ تابعی است که برای هر زوج } (id, w) \in ID \times W \text{، اهمیت واژه } w \text{ در سند با شناسه } id \text{ را نشان می‌دهد و آن را تابع (درجه) عضویت می‌نامیم؛ همچنین تعریف می‌کنیم:} \\ DB_F(w, \alpha) &= \{\langle id, \mu_F(id, w) \rangle \mid \end{aligned} \quad (1)$$

¹ Honest-but-curious

² Negligible function

³ Access pattern

⁴ Query pattern



روی مجموعه اسناد خود، یک شاخص رمزشده تولید و آن را در سمت سرور ذخیره می‌کند. کاربر می‌تواند پرسمناهای به فرم (q, α) را به دفعات درخواست بدهد. تعامل کاربر و سرور باید این امکان را برای کاربر فراهم سازد که شناسه اسنادی که میزان ارتباط آن‌ها به پرسمنان q ، پaramتر α را برآورده می‌کند، به ترتیب نزولی اهمیت به دست آورد. پرسمنان q می‌توانند تک واژه w یا یک عبارت بولی مانند $w_1 \wedge w_2$ باشد. به ازای $w = q$ خروجی پروتکل $\text{Ord}(\text{DB}_F(w, \alpha))$ در رابطه با w خواهد بود که (α) در رابطه با w تعریف شده است.

تعریف ۴ (RSSE). یک روش رمزگذاری جستجوپذیر متقارن رتبه‌بندی شده، یک دوتایی $\Pi = (\text{Setup}, \text{Search})$ است که Setup یک الگوریتم P.P.T و Search یک پروتکل است به طوری که:

▪ $(K, EDB) \leftarrow \text{Setup}(\text{DB})$ توسط کاربر اجرا می‌شود و با دریافت مجموعه سند DB، شاخص رمزشده EDB و کلید K را تولید می‌کند.

▪ $V \leftarrow \text{Search}(K, q, \alpha; EDB)$ بین کاربر و سرور اجرا می‌شود. کاربر کلید K و پرسمنان (q, α) و سرور شاخص رمزشده EDB را به عنوان ورودی‌های پروتکل تأمین می‌کند. در پایان پروتکل، کاربر یک فهرست از شناسه استناد را خروجی می‌دهد و سرور هیچ خروجی ندارد.

برای یک پروتکل RSSE شرط صحت نیز باید برقرار باشد. به این معنی که اگر کاربر و سرور پروتکل را به درستی دنبال کنند، آن‌گاه پروتکل Search برای هر پرسمنان، نتیجه درست را باز گرداند. این مفهوم در تعریف زیر دقیق شده است:

تعریف ۵ (صحت RSSE). یک RSSE دارای صحت است اگر:

$$\forall DB \forall q : \Pr \left[\begin{array}{l} (K, EDB) \leftarrow \text{Setup}(\text{DB}), \\ V \leftarrow \text{Search}(K, q, \alpha; EDB) \\ : V = \text{Ord}(\text{DB}_F(q, \alpha)) \end{array} \right] = 1$$

۱-۴- تعریف امنیت

امنیت رمزگذاری جستجوپذیر به دو صورت تطبیقی^۳ و غیر تطبیقی^۴ تعریف می‌شود. در مدل امنیت تطبیقی مهاجم این امکان را دارد که در هر زمان دلخواه، پرسمناهای خود را ارسال کند. لذا، این امکان برای مهاجم فراهم می‌شود که هر

³ Adaptive-security

⁴ Non-adaptive-security

رمزگشایی نتواند یک سیستم رمز حافظه‌ترتیب را از یک تابع اکیداً صعودی تصادفی با دامنه و برد مشابه، تمایز بدهد.

تابع شبه‌تصادفی^۱: نوعی تابع است که به نظر تصادفی می‌رسد؛ یعنی، از یک تابع که به طور یکنواخت از مجموعه تمام توابع، با دامنه و برد مشابه، انتخاب شده تمایزنپذیر محاسباتی است. به طور رسمی، تعریف زیر را داریم:

تعریف ۲ (تابع شبه-تصادفی). فرض کنید $\text{Func}[n, m]$

مجموعه کلی توابع از $\{0,1\}^n$ به $\{0,1\}^m$ باشد. یک تابع $F: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^m$ را شبه‌تصادفی نامیم اگر در زمان چندجمله‌ای، روی ورودی داده شده، قابل محاسبه و همچنین برای هر مهاجم \mathcal{A} یک تابع ناچیز negl موجود باشد به طوری که:

$$\left| \Pr \left[\mathcal{A}^{F(K,.)} (1^\lambda) = 1 : K \leftarrow \{0,1\}^\lambda \right] - \Pr \left[\mathcal{A}^{g(.)} (1^\lambda) = 1 : g \leftarrow \text{Func}[n, m] \right] \right| \leq \text{negl}(\lambda).$$

یک جایگشت شبه‌تصادفی به صورت مشابه تعریف می‌شود، با این تفاوت که F یک تابع دوسویه است و مهاجم هم به تابع و هم به وارون تابع دسترسی اوراکلی دارد.

فرضیات رمزگاری: از جمله مسائل سخت ریاضی که در این مقاله از آن استفاده خواهیم کرد، فرض سختی مسئله دیفی‌هلمن تصمیمی است. به طور شهودی این فرض بیان می‌کند که اگر g^a و g^b داده شده باشند، آن‌گاه g^{ab} تصادفی به نظر می‌رسد. این مسئله به صورت زیر تعریف می‌شود:

تعریف ۳ (فرض دیفی‌هلمن تصمیمی). گوییم مسئله DDH نسبت به \mathcal{G} سخت است اگر برای هر مهاجم \mathcal{A} یک تابع ناچیز negl موجود باشد به طوری که:

$$\left| \Pr \left[\mathcal{A} (G, q, g, g^a, g^b, g^{ab}) = 1 \right] - \Pr \left[\mathcal{A} (G, q, g, g^a, g^b, g^c) = 1 \right] \right| \leq \text{negl}(\lambda)$$

که به طور یکنواخت از $\{0\} - \mathbb{Z}_q$ انتخاب شده‌اند و (G, q, g) از اجرای $\mathcal{G}(1^\lambda)$ به دست می‌آید که G یک گروه دوری از مرتبه اول q و با مولد g است (\mathcal{G} به عنوان الگوریتم تولید گروه دوری مرتبه اول شناخته می‌شود).

۴- جستجوی رتبه‌بندی شده

یک پروتکل رمزگذاری جستجوپذیر متقارن رتبه‌بندی شده (RSSE)، یک پروتکل دو عاملی بین کاربر و سرور است. نحوه استفاده از این پروتکل به این صورت است که کاربر از

¹ Pseudo-random function

² Ranked searchable symmetric encryption

را انتخاب می‌کند؛ یعنی، $(DB; Q) \leftarrow \mathcal{A}(1^\lambda)$. سپس بازی ارسال می‌کند؛ درنهایت \mathcal{A} یک بیت باز می‌گرداند که بازی از آن به عنوان خروجی خود، که با متغیر تصادفی $\text{Simu}_{\mathcal{A}, S}^{\Sigma}(\lambda)$ نشان داده می‌شود، استفاده می‌کند.

گوییم طرح Σ ، دارای امنیت غیر تطبیقی با نشتی \mathcal{L} است، اگر برای هر مهاجم \mathcal{A} یک شبیه‌ساز P.P.T مانند \mathcal{G} وجود داشته باشد، به طوری که:

$$\Pr[\text{Real}_{\mathcal{A}}^{\Sigma}(\lambda) = 1] - \Pr[\text{Simu}_{\mathcal{A}, S}^{\Sigma}(\lambda) = 1] \leq \text{negl}(\lambda).$$

۵- بازیابی فازی برای تک واژه

هدف این بخش، ارائه یک طرح رمز RSSE است که امکان بازیابی استناد شامل تک واژه w را فراهم می‌آورد؛ به نحوی که میزان ارتباط استناد به واژه مورد نظر حداقل α باشد.

این طرح که بر اساس [8] است، از داده‌ساختارهایی مثل آرایه، فهرست پیوندی و جدول مراجعه کمک می‌گیرد و تفاوت آن با [8] در مرتب جیدن استناد و استفاده از رمزگذاری حافظترتیب است. همان‌طور که خواهیم دید، این تغییرات منجر به پشتیبانی از عملکرد مهم جستجوی رتبه‌بندی شده می‌شود. یکی از قابلیت‌های این داده‌ساختار امکان بازیابی استناد به صورت مجزا و متوالی است [8]. این ویژگی ما را قادر به پشتیبانی کارا از عملکرد جستجوی رتبه‌بندی شده می‌سازد. در ادامه، به توضیح این داده‌ساختار، توصیف قابلیت یادشده و نحوه بهره‌مندی ما از آن جهت دست‌یابی به جستجوی رتبه‌بندی شده می‌پردازیم.

برای یک آرایه A ، عنصر واقع در نشانی i را با $A[i]$ نشان می‌دهیم و برای یک عنصر x محل آن در آرایه A را با $\text{addr}_A(x)$ نمایش می‌دهیم؛ بنابراین $A[i] = x$ به این معنی است که $\text{addr}_A(x) = i$. به علاوه، یک فهرست پیوندی از عناصری n در آرایه A ذخیره شده است، دنباله‌ای از عناصرهای $N_i = (v_i, \text{addr}_A N_{i+1})$ است که $1 \leq i \leq n$ عنصر بعدی فهرست در آرایه A است. تعداد عناصر فهرست L را به وسیله $L = \#\text{نشان می‌دهیم}$. شاخص متشکل از داده‌ساختار زیر است (شکل (۱) را مشاهده کنید):

یک آرایه که برای هر $w \in W$ ، رمزشده $DB_F(w)$ در آن ذخیره شده که $DB_F(w)$ مطابق رابطه (۲) است.

یک جدول مراجعه که برای هر $w \in W$ ، اطلاعاتی برای بازیابی و رمزگشایی (w) از $DB_F(w)$ از آرایه A فراهم می‌سازد.

پرسمان را وابسته به پرسمان‌های ارسالی تا آن لحظه و پاسخ‌های دریافتی، انتخاب کند؛ اما در امنیت غیر تطبیقی مهاجم محدود به ارسال پرسمان‌ها به صورت یکجا است. بهوضوح امنیت در برابر مهاجم تطبیقی، امنیت در برابر مهاجم غیر تطبیقی را نتیجه می‌دهد؛ اما عکس این مسئله صادق نیست. از طرف دیگر، طراحی کارای پروتکل‌های امن در مدل تطبیقی دشوارتر است. مدل امنیت استفاده شده در این مقاله امنیت غیر تطبیقی است که نخستین بار در [8] مطرح شد. این مدل امنیت تضمین می‌کند که از روی پرسمان یا شاخص رمزشده، هیچ اطلاعات مفیدی که به‌طور مستقیم منجر به آشکارشدن واژه تحت پرسمان یا شاخص شود، نشست پیدا نکند. کورتمولا و همکارانش [8] برای مدل کردن مفهوم عدم نشست اطلاعات مفید در یک طرح SSE، یک تعریف امنیت دقیق ارائه کردند؛ به این صورت که، با دریافت تنها یک نشتی مشخص، یک شبیه‌ساز باید قادر باشد دید سرور را طی یک حمله غیر تطبیقی شبیه‌سازی کند. نشتی یادشده که به وسیله \mathcal{L} نشان داده می‌شود، برخی اطلاعات نه چندان حساس را مدل می‌کند. این اطلاعات شامل الگوی جستجو، الگوی دسترسی و برخی اطلاعات در مورد اندازه مجموعه استناد است؛ بنابراین تضمین می‌شود که یک سرور HBC با داشتن شاخص رمزشده EDB و دریافت پرسمان‌های متفاوت، اطلاعاتی فراتر از آن چه که در نشتی مشخص می‌شود، به دست نیاورد. تعریف ۶ (امنیت غیر تطبیقی [8]). فرض کنید Σ گوریتم حالت‌دار باشد؛ برای مهاجم \mathcal{A} و شبیه‌ساز \mathcal{G} بازی‌های زیر را تعریف می‌کنیم:

Real $_{\mathcal{A}}^{\Sigma}(\lambda)$: مهاجم \mathcal{A} مجموعه استناد DB و پرسمان‌های $Q = \{(q_1, \alpha_1); \dots; (q_l, \alpha_l)\}$ ؛ $l = \text{poly}(\lambda)$ را انتخاب می‌کند؛ یعنی، $(DB; Q) \leftarrow \mathcal{A}(1^\lambda)$. بازی الگوریتم $(K, EDB) \leftarrow \text{Setup(DB)}$ را اجرا و خروجی EDB را به مهاجم \mathcal{A} ارسال می‌کند. برای هر پرسمان $q_i \in Q$ ، $(q_i, \alpha_i) \in Q$ بازی همچنین پروتکل $(K, q_i, \alpha_i; EDB)$ را با $V \leftarrow \text{Search}(K, q_i, \alpha_i; EDB)$ ورودی (K, q_i, α_i) برای کاربر و ورودی EDB برای سرور اجرا می‌کند و رونوشتی از تعاملات و خروجی کاربر را به \mathcal{A} می‌دهد؛ درنهایت \mathcal{A} یک بیت خروجی می‌دهد که بازی از آن به عنوان خروجی خودش، که با متغیر تصادفی $\text{Real}_{\mathcal{A}}^{\Sigma}(\lambda)$ نشان داده می‌شود، استفاده می‌کند.

مهاجم \mathcal{A} مجموعه استناد DB و پرسمان‌های $Q = \{(q_1, \alpha_1); \dots; (q_l, \alpha_l)\}$ ؛ $l = \text{poly}(\lambda)$

داده‌ساختارهای A و T ، پس از تولید، در سمت سرور قرار می‌گیرند. کاربری که قصد جستجوی واژه w_i را دارد، کلید رمزگشایی و نشانی درآیه متناظر T را تولید و به همراه پارامتر $(\alpha) = \mathcal{OP}\mathcal{E}.Enc(\alpha) = \tilde{\alpha}$ به سرور ارسال می‌کند. سرور درآیه متناظر در T را پیدا و رمزگشایی می‌کند. درنتیجه، نشانی نخستین عنصر L_i در A و کلید رمزگشایی آن را بهدست می‌آورد. از طرف دیگر، چون هر عنصر حاوی یک اشاره‌گر به عنصر بعدی فهرست است، سرور می‌تواند عنصرهایی را که درجه اهمیت آن‌ها حداقل $\tilde{\alpha}$ است، به ترتیب بازیابی کند.

۱-۵- الگوریتم بازیابی فازی تک واژه (SKFR)

طرح رمز پیشنهادی (SKFR²) در الگوریتم زیر آورده شده که تفاوت‌های آن با طرح ارائه شده در [8] با رنگ قرمز مشخص شده است. الگوریتم SKFR متشکل از یک الگوریتم Setup و پروتکل Search است که آرایه A و جدول مراجعه T در طول الگوریتم Setup ساخته و پروتکل Search بین کاربر و سرور، برای جستجوی واژه مورد نظر کاربر، اجرا می‌شود.

الگوریتم SKFR

:Setup(DB)

شمارنده سراسری را $ctr = 1$ در نظر بگیر.
۱. ساخت آرایه A :

بازای هر $w_i \in W$ ، فهرست را L_i که شامل عناصر N_{ij} است به صورت زیر ساز و آن را داخل آرایه A ذخیره کن:

* بازای هر $id_j \in DB_F(w_i)$ قرار بده

$$\mu_{ij} = \mu_F(id_j, w_i)$$

* کلیدهای K_{10}, K_1, K_2, K_3 را با توزیع یکنواخت از $\{0,1\}^\lambda$ انتخاب کن.

* بازای هر $|DB_F(w_i)| \leq j \leq 1$ اعمال زیر را انجام بده:
• را j -امین شناسه در $((DB_F(w_i))$ درنظر

بگیر.

- اگر $|DB_F(w_i)| = j = |DB_F(w_i)| - k_{ij} = 1$ در غیر این صورت قرار بده ($k_{ij} \leftarrow \mathcal{SK}\mathcal{E}.Gen(1^\lambda)$
- قرار بده

$$K_0 \leftarrow \mathcal{OP}\mathcal{E}.Gen(1^\lambda)$$

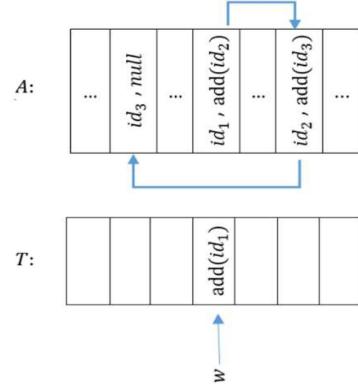
$$\tilde{\mu}_{ij} = \mathcal{OP}\mathcal{E}.Enc(K_0, \mu_{ij})$$

و

$$v_{ij} = \langle id_{ij}, \tilde{\mu}_{ij}, k_{ij} \rangle$$

² Single keyword fuzzy retrieval

به ازای هر $w_i \in W$ ، یک فهرست پیوندی L_i تولید می‌کنیم که هر عنصر آن به ترتیب یک عضو $\langle id, \mu_F(id, w_i) \rangle$ از $Ord(DB_F(w_i))$ را شامل می‌شود؛ سپس، کلیه عنصرهای تمام فهرست‌ها در آرایه A با یک ترتیب تصادفی و به صورت رمزشده ذخیره می‌شوند. قسمت (...) $\mu_F(\dots)$ هر عنصر با یک روش رمزگذاری حافظه‌ترتیب و بخش دیگر با روش رمزگذاری معمول رمز می‌شوند. قبل از رمزگردان عنصر j -ام از فهرست L_i ، یک اشاره‌گر (نسبت به آرایه A) به عنصر $(j+1)$ -ام فهرست و یک کلید برای رمزگردان آن، به عنصر j -ام اضافه می‌شود. به این ترتیب، با داشتن محل نخستین عنصر L_i در آرایه A و کلید رمزگشایی آن، سرور قادر است، سایر عنصرهای فهرست L_i را بازیابی کند. برای تعیین محل و رمزگشایی T نخستین عنصر هر فهرست در آرایه، از جدول مراجعه $w_i \in W$ استفاده می‌شود. هر درآیه در جدول T متناظر با یک تابع $value$ محل و کلید رمزگشایی نخستین عنصر L_i در A را شامل می‌شود. خود $value$ بهوسیله یک تابع $address$ برای شبه‌تصادفی¹ (PRF) رمز می‌شود. مؤلفه $address$ برای تعیین محل درآیه‌های T به کار می‌رود.



(شکل-۱): فرض کنید واژه w در ۳ سند id_1, id_2 و id_3 واقع شده است. از طریق جدول T شناسه و نشانی نخستین سند شامل w یعنی id_1 را در آرایه A بهدست می‌آوریم؛ سپس، از طریق پیوندهای متوالی شناسه‌های id_1 و id_2 و بازیابی می‌شوند. توجه داریم که در این شکل، جهت سهولت، اطلاعات به صورت رمزنشده نمایش داده شده‌اند.

(Figure-1): Suppose that the keyword w is located in three documents with identifiers id_1, id_2 and id_3 . From table T , we get the ID and address of the first document containing w , i.e. id_1 in array A . Then, id_2 and id_3 are retrieved by sequential links. Note that in this figure, for convenience, the information is displayed as plain.

¹ Pseudo-random function



SKFR صحیح و دارای امنیت غیر تطبیقی با نشتی \mathcal{L} است (که \mathcal{L} به عنوان ورودی شبیه‌ساز در روند اثبات مشخص شده است).

اثبات: بازی شبیه‌سازی شده در تعریف ۶ به صورت زیر است:

۱. مهاجم \mathcal{A} مجموعه اسناد DB و پرسمان‌های $Q = \{(w_1, \alpha_1); (w_l, \alpha_l): l = \text{Poly}(\lambda)\}$

را انتخاب می‌کند، یعنی، $(DB; Q) \leftarrow \mathcal{A}(1^\lambda)$.

۲. شبیه‌ساز \mathcal{K} به عنوان ورودی، اندازه واژنامه، اندازه مجموعه‌استاند (یعنی $(\sum_{w \in W} |\text{DB}(w)|)$) و همچنین الگوی دسترسی $\text{Ord}(\text{DB}_F(w_i, \alpha_i))$ و الگوی پرسمان را به ازای هر w_i دریافت می‌کند.

۳. شبیه‌ساز، اعداد تصادفی اما نزولی μ_{ij} را انتخاب $\text{Ord}(\text{DB}_F(w_i, \alpha_i))$ به اعضای (w_i) را می‌کند و به ترتیب، به اعضای (w_i) نسبت می‌دهد.

۴. به ازای هر پرسمان w_i ، شبیه‌ساز مشابه الگوریتم واقعی و با استفاده از ورودی‌های خود قادر است، سلول‌های متناظر A را شبیه‌سازی کند. در واقع، شبیه‌ساز \mathcal{K} در شبیه‌سازی آرایه A نیازی به w_i ندارد و داشتن فهرست $\pi(w_i, \alpha_i)$ $\text{Ord}(\text{DB}_F(w_i, \alpha_i))$ به ازای واژه‌های موجود در پرسمان‌ها برای او کفایت می‌کند. شبیه‌ساز سایر سلول‌های خالی A را با مقادیر تصادفی پر می‌کند.

۵. به ازای هر پرسمان w_i ، شبیه‌ساز دو عدد تصادفی γ_i و η_i را به ترتیب از برد π و f انتخاب و مانند الگوریتم واقعی جدول T را شبیه‌سازی می‌کند (در واقع چون w_i را ندارد به جای (K_3, w_i) و $\pi(K_3, w_i)$ دو عدد $f(K_2, w_i)$ و $\pi(K_2, w_i)$ تصادفی انتخاب می‌کند).

شبیه‌ساز سایر سلول‌های خالی جدول T را با مقادیر تصادفی پر می‌کند.

۶. شبیه‌ساز داده ساختارهای A و T را خروجی می‌دهد. همچنین به ازای هر پرسمان w_i ، شبیه‌ساز مقدار $t_i = (\gamma_i, \eta_i)$ متناظر آن از مرحله قبل و همچنین عدد $\tilde{\alpha}$ را خروجی می‌دهد به طوری که $\tilde{\alpha}$ از تمام مقادیر j μ_j در مرحله ۳، کوچک‌تر باشد.

با توجه به فرضیات قضیه، واضح است که بازی شبیه‌سازی بالا قابل تمایز از بازی واقعی نیست.

۶- بازیابی فازی برای عبارت بولی

روش ارائه شده در این بخش تعمیمی از روش کش و همکارانش در [6] است و هدف آن بازیابی فازی اسنادی است که به میزان کمینه α به عبارت بولی:

- اگر $|\text{DB}_F(w_i)| = j$ قرار بده $= \text{addr}_A(N_{i(j+1)})$ در غیر این صورت: $NULL$

$\text{addr}_A(N_{i(j+1)}) = \phi(K_1, ctr + 1)$

- قرار بده $\langle v_{ij}, \text{addr}_A(N_{i(j+1)}) \rangle$

- عنصر N_{ij} را با کلید $k_{i(j-1)}$ رمز کن و آن را درون آرایه A قرار بده به طوری که:

$A[\phi(K_1, ctr)] \leftarrow \text{SK}\mathcal{E}.\text{Enc}(k_{i(j-1)}, N_{ij})$

- شمارنده ctr را یک واحد افزایش بده.

۷. ساخت جدول مراجعه T :

به ازای هر $w_i \in W$ قرار بده

$T[\pi(K_3, w_i)] = (\text{addr}_A(N_{i1}) || k_{i0}) \oplus f(K_2, w_i)$

۳. خروجی:

مقادیر $EDB = (A, T)$ و $K = (K_0, K_1, K_2, K_3)$ را برگردان.

پروتکل :Search(K, w, α, EDB)

۱. کاربر: پیام $\langle \pi(K_3, w), f(K_2, w) \rangle$ و $t = \langle \pi(K_3, w), f(K_2, w) \rangle$

۲. سرور: اعمال زیر را انجام می‌دهد؛

* t را به صورت $\langle \gamma, \eta \rangle$ در نظر می‌گیرد و قرار می‌دهد.

. $\theta = T[\gamma]$

* اگر $\mathcal{L} \neq \theta$ ، آنگاه $\theta \oplus \eta$ را به صورت $\langle a_1, k_0 \rangle$ بازنویسی می‌کند، قرار می‌دهد $1 = j$ و ادامه می‌دهد.

در غیر این صورت نماد \perp را بر می‌گرداند.

* تا زمانی که $a_j \neq NULL$ اعمال زیر را انجام می‌دهد؛

* $N_j = \text{SK}\mathcal{E}.\text{Dec}(k_{j-1}, A[a_j])$

• N_j را به شکل $\langle v_j, a_{j+1} \rangle$ بازنویسی می‌کند که $v_j = \langle id_j, \tilde{\mu}_j, k_j \rangle$

• اگر $\tilde{\mu}_j \geq \tilde{\alpha}$ ، آنگاه شناسه id_j را به کاربر ارسال می‌کند. در غیر این صورت توقف می‌کند و ادامه نمی‌دهد.

• شمارنده زیرا یک واحد افزایش می‌دهد.

۲-۵- تحلیل امنیت الگوریتم SKFR

در این بخش به تحلیل امنیت الگوریتم پیشنهادی می‌پردازیم و نشان می‌دهیم این طرح دارای امنیت اثبات‌پذیر در مدل شبیه‌سازی است.

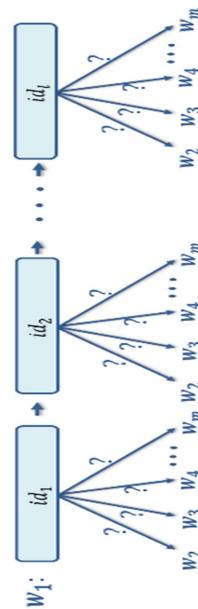
قضیه ۱. فرض کنید OPE یک طرح رمز حافظه‌ترتبی امن، $SK\mathcal{E}$ یک طرح رمز CPA¹ امن، ϕ و π دو PRP² امن و f یک PRF امن باشد. آن‌گاه طرح رمز ارائه شده در الگوریتم

¹ Chosen-plaintext attack

² Pseudo-random permutation

(id_j, w_i) سرور و کاربر نیاز به همکاری و تعامل با یکدیگر دارند. بازیابی μ_{ij} را می‌توان با استفاده از یک روش ابتدایی، اما ناکارا به صورت زیر انجام داد. سرور هرشناسه بازیابی شده id_j را به کاربر بازمی‌گرداند، سپس کاربر یک نشان مبتنی بر زوج (id_j, w_i) می‌سازد و به سرور ارسال می‌کند. سرور با استفاده از یک داده‌ساختار $XSet$ که توسط زوج‌های عضو $ID \times W$ شاخص‌گذاری و در مقابل در سمت سرور بارگذاری شده است و همچنین با استفاده از نشان دریافتی از طرف کاربر مقدار μ_{ij} را از داده‌ساختار $XSet$ بازیابی می‌کند. واضح است که این روش به طور کامل ناکارا است و نیاز به تعامل با کاربر دارد. در حالی که کاربردهای موجود در رایانش ابری همواره سعی دارند از تعامل با کاربر اجتناب شود. کش و همکارانش [6] روشی هوشمندانه را به منظور حذف این تعامل ارائه کردند. در روش آن‌ها در حقیقت $XSet$ به عنوان یک مجموعه از عناصر تصادفی شبیه‌سازی می‌شود، به طوی که اگر $id_j \in DB(w_i)$ ، آن‌گاه عنصر استفاده از توابع شبه‌تصادفی، عناصر موجود در $XSet$ به‌ نحوی ساخته می‌شود که شامل دو بخش هستند. یک بخش فقط شامل شناسه و بخش دیگر تنها شامل واژه است. بخشی که توسط کاربر تولید می‌شود و فقط شامل واژه است در حقیقت عضوی از یک گروه دوری است؛ به‌ نحوی که مسئله دیفی‌هلمن تصمیمی در این گروه سخت است و لذا مقادیر متعدد تولید شده توسط کاربر اطلاعاتی را در اختیار سرور قرار نمی‌دهد. به این ترتیب کاربر و سرور بدون آشکارکردن واژه یا انجام تعاملات بیشتر قادر به حل مشکل یادشده خواهند بود. در زیر به توصیف روش پیشنهادی خود می‌پردازیم که به طور ضمنی از روش کش و همکارانش بهره می‌برد. از آنجا که روش پیشنهادی کش و همکارانش جستجوی رتبه‌بندی شده را پشتیبانی نمی‌کند، داده‌ساختار $XSet$ در روش آن‌ها ساده‌تر است و تنها بررسی عضویت واژه در سند مورد نظر را ممکن می‌سازد. داده‌ساختار پیشنهادی آن‌ها همچنین دارای اندازه‌ای از مرتبه $O(N)$ و زمان جستجوی $O(1)$ است که $N = \sum_{w \in W} DB(w)$ است؛ در حالی که داده‌ساختار $XSet$ در روش ما میزان ارتباط واژه به سند را نیز ذخیره می‌کند. گفتنی است که دست‌یابی به این عملکرد در روش ما، به قیمت افزایش فضای لازم جهت ذخیره‌سازی داده‌ساختار $XSet$ به دست آمده است؛ به‌ نحوی که این داده‌ساختار در روش پیشنهادی ما دارای اندازه $(|W| \times |ID|)$ و زمان جستجوی $O(1)$ است.

$q = (w_1 \wedge w_2 \wedge \dots \wedge w_m)$ مربوط هستند (الگوریتم BooleanFR). فرض کنید واژه w_1 واژه با کمترین بسامد باشد، به این معنی که تعداد اسنادی که میزان ارتباط آن‌ها به واژه w_1 حداقل α است، کمترین تعداد باشد (در مقایسه با بقیه واژه‌های موجود در پرسمان) برای جستجوی رتبه‌بندی شده پرسمان عطفی q در حالت رمزنشده، ابتدا شناسه اسنادی را که میزان ارتباط آن‌ها با واژه w_1 حداقل α است، بازیابی، سپس به‌ازای هر شناسه بازیابی شده id_j بررسی می‌کنیم که آیا این شناسه واژه w_i را برای $i = 2, \dots, m$ شامل می‌شود و اگر شامل می‌شود میزان ارتباط شناسه یادشده به این واژه چقدر است که این مقدار را با μ_{ij} نشان می‌دهیم؛ درنهایت اگر داشته باشیم $m_j \geq \alpha$ که $m_j = \min\{\mu_{ij}: i = 1, \dots, m\}$ ، به این معنی است که شناسه id_j پرسمان q را برآورده می‌کند.



(شکل-۲): الگوریتم اولیه برای جستجوی پرسمان عطفی در حالت رمزنشده.

(Figure-2): The basic algorithm for the conjunctive search query in the plain mode.

شکل (۲) الگوریتم جستجوی برای پرسمان عطفی را در حالت رمزنشده نشان می‌دهد؛ اما تعمیم این روش برای داده‌های رمزشده به طور بدیهی امکان‌پذیر نیست. یک تعمیم از این روش برای داده‌های رمزشده به صورت زیر است. ابتدا با استفاده از یک روش جستجوی رتبه‌بندی شده برای تک‌واژه (مانند الگوریتم SKFR) شناسه‌هایی که واژه w_1 را شامل می‌شوند و دست‌کم به میزان α با آن ارتباط دارند، در سمت سرور بازیابی می‌شوند. از طرف دیگر، واژه‌های فقط در اختیار کاربر هستند؛ لذا به منظور بازیابی μ_{ij} متناظر با زوج

می‌تواند استاد شامل پرسمن (w_1, w'_2) را بازیابی کند. برای رفع این مشکل، کش و همکارانش پیشنهاد دادند که $F_p(K_I, id_j)$ را به صورت کورشده سمت سرور ذخیره کنیم؛ $F_p(K_I, id_j) \cdot F_p(K_I, id_j)$ مقدار \cdot معنی بهجای $F_p(K_I, id_j)$ در سمت سرور قرار داده می‌شود و متعاقباً کاربر بهجای $g^{F_p(K_X, w_i)}$ عبارت $g^{F_p(K_I, id_j) \cdot F_p(K_X, w_i)}$ را به سرور ارسال می‌کند (خواننده علاقمند می‌تواند به [6] مراجعه کند). درنهایت با ترکیب ایده ما و ایده موجود در [6]، الگوریتم مورد نظر (BooleanFR²) به صورت زیر به دست می‌آید که تفاوت آن با الگوریتم SKFR با رنگ قرمز مشخص شده است.

الگوریتم BooleanFR

الگوریتم Setup(DB)

شمارنده سراسری را $ctr = 1$ در نظر بگیر.

۱. ساخت آرایه A:

به ازای هر $w_i \in W$ ، لیست L_i را که شامل عناصر N_{ij} است به صورت زیر بساز و آن را داخل آرایه A ذخیره کن.

* به ازای هر $id_j \in DB_F(w_i)$ قرار بده

$$\mu_{ij} = \mu_F(id_j, w_i)$$

* کلیدهای $K_{i0}, K_1, K_2, K_3, K'$ را با توزیع یکنواخت از $\{0,1\}^\lambda$ انتخاب کن.

* به ازای هر $j \leq |DB_F(w_i)| - 1$ اعمال زیر را انجام بده:

• را j -امین شناسه در $Ord(DB_F(w_i))$ در نظر بگیر.

• اگر $k_{ij} = j = |DB_F(w_i)| - 1$ قرار بده $k_{ij} \leftarrow 1$

صورت قرار بده $k_{ij} \leftarrow SK\mathcal{E}.Gen(1^\lambda)$

• قرار بده

$$K_0 \leftarrow \mathcal{OP}\mathcal{E}.Gen(1^\lambda)$$

$$\tilde{\mu}_{ij} = \mathcal{OP}\mathcal{E}.Enc(K_0, \mu_{ij})$$

$$xid_{ij} = F_p(K_I, id_{ij})$$

$$z_{ij} = F_p(K_z, w_i || j)$$

$$y_{ij} = xid_{ij} \cdot z_{ij}^{-1}$$

$$\tilde{id}_{ij} = SK\mathcal{E}.Enc'(K', id_{ij})$$

و

$$v_{ij} = (\tilde{id}_{ij}, \tilde{\mu}_{ij}, k_{ij}, y_{ij})$$

• اگر $j = |DB_F(w_i)|$ قرار بده

 دربیگر این صورت:

$$addr_A(N_{i(j+1)}) = \phi(K_1, ctr + 1)$$

 قرار بده $((v_{ij}, addr_A(N_{i(j+1)}))$

 عنصر N_{ij} را با کلید $k_{i(j-1)}$ رمز کن و آن را درون

 آرایه A قرار بده، به طوری که:

² Boolean fuzzy retrieval

۶-۱- الگوریتم بازیابی فازی عبارت بولی

فرض کنید G گروهی دوری با مولد g و از مرتبه نخست p باشد و F_p یک PRF باشد که برد آن در \mathbb{Z}_p واقع است.

ایده الگوریتم به این صورت است که کاربر در ابتدای داده‌ساختار $XSet$ را سمت سرور ذخیره می‌کند به طوری که متناظر با هر زوج $(id_j, w_i) \in ID \times W$ سلول شامل مقدار μ_{ij} موجود باشد. داده‌ساختارهای A و T نیز شبیه به داده‌ساختارهای مشابه در الگوریتم SKFR تولید و بر روی رسانه ذخیره‌سازی سرور قرار داده می‌شوند. جهت جستجوی پرسمن ($w_1 \wedge w_2 \wedge \dots \wedge w_m$)، ابتدا مشابه روش قبل، کاربر درخواستی برای جستجوی واژه w_1 به سرور ارسال و سرور استاد با کمینه‌های ارتباط α به واژه w_1 را بازیابی می‌کند (یعنی $Ord(DB_F(w_1, \alpha))$). برای هر سند بازیابی شده id_j (که عضوی از $Ord(DB_F(w_1, \alpha))$ است) کاربر و سرور با همکاری هم، نشانی‌های $g^{F_p(K_I, id_j) \cdot F_p(K_X, w_i)}$ ، $i = 2, \dots, m$ از داده‌ساختار $XSet$ را به دست می‌آورند؛ به طوری که برای هر $i = 2, \dots, m$ میزان ارتباط واژه w_i به سند id_j به صورت μ_{ij} در این محل ذخیره شده است. با توجه به قوانین فازی در رابطه (۳)، ارتباط سند id_j به عبارت بولی $(w_1 \wedge w_2 \wedge \dots \wedge w_m)$ به صورت: $m_j = \min\{\mu_{ij}; i = 1, \dots, m\}$ درنهایت با دردست داشتن m_j ، مرتبطترین استاد به عبارت بولی، به ترتیب به دست می‌آیند. نکته‌ای که باید مورد توجه قرار داد، این است که استاد بازیابی شده، یعنی $Ord(DB_F(w_1, \alpha))$ ، سمت سرور هستند و کاربر به آن‌ها دسترسی ندارد؛ بنابراین، برای محاسبه نشانی $g^{F_p(K_X, w_i)}$ ، کاربر باید مقدار $g^{F_p(K_I, id_j) \cdot F_p(K_X, w_i)}$ را برای هر $i = 2, \dots, m$ به سرور ارسال کند و سرور با در دست داشتن $F_p(K_I, id_j)$ (که از قبل در داده‌ساختار A سمت سرور ذخیره شده است)، نشانی مورد نظر را به دست می‌آورد؛ اما این روش نیز منجر به مشکلات امنیتی می‌شود، زیرا سرور می‌تواند با دریافت اطلاعات مربوط به پرسمن‌های قبلی، برای پرسمن جدیدی که سؤال نشده است، استاد مربوطه را بازیابی کند. به عنوان مثال با استفاده از اطلاعات دریافتی برای پرسمن‌های دریافتی (w'_1, w'_2) و (w_1, w_2) در

¹ توجه داریم که مقادیر $g^{F_p(K_I, id_j) \cdot F_p(K_X, w_i)}$ در روش کش و همکارانش به عنوان اعضای مجموعه $XSet$ لحاظ شدن، ولی در روش ما به عنوان نشانی سلول‌های داده‌ساختار $XSet$ هستند.



- در غیر این صورت پیام STOP را به کاربر ارسال می‌کند و همین جا متوقف می‌شود.
- شمارنده زرایک واحد افزایش می‌دهد.
- * مجموعه V را به طور نزولی بر حسب m_j -ها مرتب می‌کند و سپس این مجموعه مرتب را بر می‌گرداند.
- مشاهده می‌شود که الگوریتم بالا به راحتی قابل تعمیم به هر پرسمنان بولی به صورت $(w_1 \wedge \dots \wedge w_m) \psi(w_2, \dots, w_m)$ است که ψ یک فرمولی بولی دلخواه است.

۲-۶- تحلیل امنیت الگوریتم BooleanFR

حال به تحلیل امنیت الگوریتم پیشنهادی خود می‌پردازم. نشان می‌دهیم الگوریتم BooleanFR دارای امنیت بر مبنای شبیه‌سازی است.

قضیه ۱. فرض کنید شرایط قضیه ۱ و همچنین F_p یک PRF امن و فرض DDH در گروه G برقرار باشد. آن‌گاه طرح رمز ارائه شده در الگوریتم‌های BooleanFR دارای امنیت غیر تطبیقی با نشتی L است (که نشتی L در طول اثبات به عنوان ورودی شبیه‌ساز مشخص شده است).

اثبات: در اینجا ما طرح کلی اثبات را ارائه می‌دهیم. برای اطلاع از جزئیات اثبات (به خصوص در مورد نشتی الگوریتم و شبیه‌سازی $XSet$) خواننده علاقمند می‌تواند به [6] مراجعه کند و با استدلال مشابه از روند جزئیات مطلع شود.

۱. مهاجم \mathcal{A} مجموعه اسناد DB و پرسمنان های

$$Q = \{(q_1, \alpha_1); l = \text{poly}(\lambda)\} \dots ; (q_l, \alpha_l); Q \leftarrow \mathcal{A}(1^\lambda)$$

۲. واضح است که شبیه‌ساز \mathcal{S} ، به عنوان ورودی اندازه واژه‌نامه، اندازه مجموعه اسناد (یعنی $|ID| \times W$) و همچنین $|\sum_{w \in W} |\text{DB}(w)||$ ، اندازه الگوی دسترسی یعنی $|\text{Ord}(\text{DB}_F(w_1, \alpha))|$ ، اندازه الگوی دسترسی یعنی $|\text{Ord}(\text{DB}_F(q, \alpha))|$ و الگوی پرسمنان را به ازای مؤلفه نخست پرسمنان (یعنی w_1) دریافت می‌کند. فرض کنیم مهاجم دو پرسمنان زیر را سؤال کرده باشد.

$$q = (w_1 \wedge w_2 \wedge \dots \wedge w_m)$$

$$q' = (w'_1 \wedge w'_2 \wedge \dots \wedge w'_m)$$

به طوری که $w_2 = w'_3$ و

$$id \in \text{Ord}(\text{DB}_F(w_1, \alpha)) \cap \text{Ord}(\text{DB}_F(w'_1, \alpha')).$$

در این صورت نشتی که شبیه‌ساز به عنوان ورودی دریافت می‌کند؛ به صورت زیر است:

شبیه‌ساز می‌داند دومین مؤلفه در q با سومین مؤلفه در q' مساوی است. همچنین، اطلاعاتی در مورد

$$A[\phi(K_1, ctr)] \leftarrow \mathcal{SK}\mathcal{E}.\text{Enc}(k_{i(j-1)}, N_{ij})$$

• شمارنده ctr را یک واحد افزایش بدهد.

۲. ساخت جدول مراجعة

به ازای هر $w_i \in W$ قرار بده

$$T[\pi(K_3, w_i)] = \langle \text{addr}_A(N_{i1}) || k_{i0} \rangle \oplus f(K_2, w_i)$$

۳. ساخت جدول مراجعة

به ازای هر $(w, id) \in W \times ID$ قرار بده

$$XSet[g^{F_p(K_X, w).cid}] = \tilde{\mu}_F(id, w)$$

۴. خروجی:

مقادیر $K = (K_1, K_Z, K_X, K_0, K_1, K_3)$ و $(A, T, XSet)$ را برگردان.

پروتکل :Search(K, q, α, EDB)

۱. کاربر: پیام‌های زیر را به سرور ارسال می‌کند؛

$$t = (\pi(K_3, w), f(K_2, w))$$

$$\tilde{\alpha} = \mathcal{OPE}.\text{Enc}(K_0, \alpha)$$

$$(xtoken[1], xtoken[2], \dots)$$

که $xtoken[k]$ به صورت زیر تعریف می‌شود؛

به ازای ... $j = 1, 2, \dots$ و تا زمانی که سرور پیام STOP دهد

اعمال زیر را انجام می‌دهد

- به ازای $i = 2, \dots, m$ قرار می‌دهد

$$xtoken[i, j] \leftarrow g^{F_p(K_Z, w_1 || j) \cdot F_p(K_X, w_i)}$$

- در نهایت قرار می‌دهد

$$xtoken[j] = xtoken[2, j], \dots, xtoken[m, j]$$

۲. سرور: اعمال زیر را انجام می‌دهد؛

* t را به صورت $\langle \gamma, \eta \rangle$ در نظر می‌گیرد و قرار می‌دهد

$$\theta = T[\gamma]$$

* اگر $\perp \neq \theta$. آن‌گاه $\theta \oplus \eta$ را به صورت $\langle a_1, k_0 \rangle$ بازنویسی

می‌کند، قرار می‌دهد $j = 1, \dots, m$ و ادامه می‌دهد.

در غیر این صورت نماد \perp را بر می‌گرداند.

* تا زمانی که $a_j \neq NULL$ اعمال زیر را انجام می‌دهد؛

$$N_j = \mathcal{SK}\mathcal{E}.\text{Dec}(k_{j-1}, A[a_j])$$

* N_j را به شکل $\langle v_j, a_{j+1} \rangle$ بازنویسی می‌کند که

$$v_j = \langle \tilde{id}_{1j}, \tilde{\mu}_{1j}, k_{1j}, y_{1j} \rangle$$

* اگر $\tilde{\alpha} \geq \tilde{\mu}_{1j}$ آن‌گاه به ازای $i = 2, \dots, m$ اعمال زیر

را انجام می‌دهد؛

- مقادیر $\tilde{\mu}_{ij} = XSet[xtoken[i, j]^{\gamma^1_j}]$ را بازیابی

می‌کند. سپس مقدار زیر را محاسبه می‌کند

$$m_j = \min\{\tilde{\mu}_{ij}; i = 1, 2, \dots, m\}$$

- اگر $m_j \geq \tilde{\alpha}$ آن‌گاه مجموعه V را به صورت

$$V = V \cup \{(i\tilde{id}_{1j}, m_j)\}$$



با توجه به فرضیات قضیه، بازی شبیه‌سازی بالا از بازی واقعی تمایزنپذیر است.

۷- نتیجه‌گیری

در این مقاله نشان داده شد که با استفاده از قابلیت‌هایی که داده‌ساختارها فراهم می‌کنند، می‌توان عملکرد طرح‌های موجود را افزایش داد. از جمله عملکرد جستجوی رتبه‌بندی شده مورد بررسی قرار گرفت. بازیابی فازی به عنوان روشی کارا در حوزه بازیابی اطلاعات برای جستجوی رتبه‌بندی شده روی داده‌های رمزشده به کار برده شد. در جدول (۱) روش‌های پیشنهادی خود (SKFR و BooleanFR) را با روش‌های مشابه (جستجوی پذیر BooleanFR) مقایسه کردیم. همان‌طور که در این جدول مشاهده می‌شود، طرح ما، برای پشتیبانی از پرسمان‌های تکوازه‌ای، نسبت به طرح‌های [21] و [23] به طور قابل توجهی کارتر است؛ همچنین، مقایسه طرح پیشنهادی ما با طرح‌های مشابه [4] و [22]، برای پشتیبانی از پرسمان‌های بولی، کارایی آن نسبت به طرح‌های یادشده را نشان می‌دهد.

	our BooleanFR	our SKFR	$O(t_{ope}, N)$	$O(t_{ope}, W d)$	$O(\alpha)$	$O(\alpha)$	Boolean	yes
Wang et.al. [21]	Zhang et.al. [23]	Cao et.al. [4]	Xia et.al. [22]					
$O(d, W , t_{ope})$	$O(t_{ope}, t_{exp}, N)$	$O(d, W ^2)$	$O(d, W ^2)$	$O(W \log d + k)$	$O(\alpha)$	$O(1)$	SKS	yes
$O(d)$				$O(W ^2)$	$O(W ^2)$	Boolean		
$O(1)$						SKS		yes

Ord(DB_F(w₁, α) ∩ Ord(DB_F(w'₁, α')) نیز به دست

می‌آورد از جمله ناتهی بودن آن [6].

۳. شبیه‌سازی داده‌ساختارهای A و T، به ازای مقادیر

Mشتراك در دو طرح الگوریتم BooleanFR و SKFR مشابه قضیه ۱ انجام می‌شود. به ازای مقادیر جدیدی که در این طرح نسبت به طرح الگوریتم SKFR داریم، یعنی مقادیر $i_{id_{ij}}$ ، $\tilde{i}_{id_{ij}}$ ، z_{ij} و Enc'_{F_p} انتخاب می‌شوند.

۴. شبیه‌سازی داده‌ساختار XSet: ابتدا شبیه‌ساز به ازای

هر $w \in W$ مقدار تصادفی f از برد F_p را انتخاب می‌کند (در واقع چون شبیه‌ساز w را ندارد به جای

$F_p(K_X, w)$ مقدار تصادفی انتخاب می‌کند).

حال به ازای هر پرسمان $q = w_1 \wedge \dots \wedge w_m$

سلول‌های متناظر به صورت زیر شبیه‌سازی می‌شوند:

به ازای هر واژه w_i که $i = 2, \dots, m$ ، مقادیر

مذکور در بالا استخراج می‌شوند.

شبیه‌ساز همچنین مقادیر xid را (متناظر با w_1 و

اعضای ((Ord(DB_F(w₁))) که در مرحله ۳ محاسبه

کرده است، استخراج و سلول‌های مورد نظر را مطابق

الگوریتم واقعی می‌سازد؛ به طوریکه مقادیر $i_{id_{ij}}$ مورد

نیاز به طور تصادفی انتخاب شده‌اند. در پایان سلول‌های

خالی با مقادیر تصادفی پر می‌شوند.

۵. شبیه‌سازی مقادیر $[i, j]$: می‌دانیم در الگوریتم

واقعی داریم:

$xtoken[i, j] \leftarrow g^{F_p(K_Z, w_1 || j) \cdot F_p(K_X, w_i)}$

شبیه‌ساز به جای مقادیر

$F_p(K_X, w_i)$ ، $xid_{1j} = F_p(K_Z, w_1 || j)$

مقادیر متناظر را که در مراحل ۲ و ۳ محاسبه کرده است، در

نظر می‌گیرد و $[j]$ را مطابق الگوریتم واقعی محاسبه

می‌کند.

۶. شبیه‌سازی نشان t و مقدار $\tilde{\alpha}$ مانند قضیه ۱ انجام

می‌شود؛ به طوری که $\tilde{\alpha} \geq \tilde{\mu}_{1j}$ به ازای:

$j = 1, \dots, |\text{Ord(DB}_F(w_1, \alpha)|$

و همچنین، به تعداد $|\text{Ord(DB}_F(q, \alpha)|$ از این اندیس‌های

j در نامساوی $m_j \geq \tilde{\alpha}$ صدق می‌کنند که m_j مطابق

الگوریتم واقعی محاسبه می‌شود.

درواقع مقادیر $i_{id_{ij}}$ طوری انتخاب می‌شوند که شرایط بالا

روی $\tilde{\alpha}$ امکان پذیر باشند.

۷. شبیه‌ساز داده‌ساختارهای A و T و همچنین

نشان t، $\tilde{\alpha}$ و $xtoken[j]$ را خروجی می‌دهد.

- 2013 USA. Proceedings, Part I, pp. 353–373, 2013.
- [7] S. Cui, M. Asghar, S. Galbraith, G. Russello, ObliviousDB, “Practical and Efficient Searchable Encryption with Controllable Leakage”, *FPS 2017*, pp. 189-205, 2017.
- [8] R. Curtmola, J.A. Garay, S. Kamara, R.Ostrovsky, “Searchable symmetric encryption: improved definitions and efficient constructions”, *CCS 2006*, USA, pp. 79–88, 2006.
- [9] E. Etemad, A. Küpcü, C. Papamanthou, D. Evans, “Efficient Dynamic Searchable Encryption with Forward Privacy”, *PoPETs 2018*, vol.1, pp. 5-20, 2018.
- [10] E.Goh, “secure indexes”, IACR Cryptology ePrint Archive 2003, pp. 216, 2003.
- [11] S. Kamara, T. Moataz, Boolean, “searchable symmetric encryption with worst-case sub-linear complexity”, *EUROCRYPT*, vol. 3, pp. 94-124 2017.
- [12] S. Kamara, T. Moataz, *SQL on Structurally-Encrypted Databases*, IACR Cryptology ePrint Archive, pp. 453, 2016.
- [13] S. Kamara, C. Papamanthou, *Parallel and dynamic searchable symmetric encryption*, Japan. pp. 258–274, 2013.
- [14] S. Kamara, C. Papamanthou, T. Roeder, *Dynamic searchable symmetric encryption*, CCS 2012, USA. pp. 965–976, 2012.
- [15] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, W. Lou, *Fuzzy keyword search over encrypted data in cloud computing*, INFOCOM 2010, USA, pp. 441–445 , 2010.
- [16] X. Phuong, L. Ngoc, *Efficient Secure Text Retrieval on Multi-Keyword Search*, SoICT 2017, pp. 270-277, 2017.
- [17] T. Radecki, *Fuzzy set theoretical approach to document retrieval*, Inf. Process. Manage, vol. 15(5), pp. 247–259, 1979.
- [18] D.X. Song, D. Wagner, A. Perrig, “Practical techniques for searches on encrypted data”, *IEEE SP 2000*, USA, pp. 44–55, 2000.
- [19] E. Stefanov, C. Papamanthou, E. Shi, *Practical dynamic searchable encryption with small leakage*, IACR Cryptology ePrint Archive 2013, pp.832, 2013.
- [20] M. Strizhov, I. Ray, *Multi-keyword Similarity Search over Encrypted Cloud Data*, SEC 2014: ICT Systems Security and Privacy Protection. Berlin. Vol. 428, pp. 52-65, 2014.
- [21] C. Wang, N. Cao, K. Ren, W. Lou, “Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data”, *IEEE Trans. Parallel Distrib. Syst*, vol. 23(8), pp. 1467-1479, 2012.

Scheme	Cash et.al [6]	Curimula et.al [8]	$O(N)$	$O(\text{DB}(w))$	$O(1)$	Boolean	SKS	no
Setup			$O(N)$	$O(\text{DB}(w))$	$O(1)$			
Search-server								
Search-client								
Query								
Ranked search								

(جدول-۱): مقایسه با کارهای مشابه

(Table-1): Comparison with related works

در این جدول، $|W|$ نشان‌دهنده اندازه و اسناد، d تعداد کل اسناد، پارامتر k تعداد فایل‌های مورد نظر کاربر برای بازیابی، $t_{pair}, t_{exp}, t_{ope}$ به ترتیب مدت زمان رمزگذاری OPE، توان رسانی در گروه دوری و انجام عمل جورسازی هستند؛ همچنین، پارامتر α کمیت مورد نظر کاربر برای ارتباط سند به واژه است و N اندازه مجموعه اسناد است، یعنی، $\sum_{w \in W} \text{DB}(w)$

۸- References - مراجع

- [1] F. Baldimtsi, O. Ohrimenko, “Sorting and Searching Behind the Curtain,” *Lecture Notes in Computer Science*, FC 2015, Berlin. No.8975. pp. 127-146, 2015.
- [2] A. Boldyreva, N. Chenette, “Efficient fuzzy search on encrypted data”, *FSE 2014*, UK. pp. 613–633, 2014.
- [3] R. Bost, B. Minaud, O. Ohrimenko, “Forward and Backward Private Searchable Encryption from Constrained Cryptographic Primitives”, pp. 1465-1482, 2017.
- [4] N. Cao, C. Wang, M. Li, K. Ren, W. Lou, “Privacy-preserving multi-keyword ranked search over encrypted cloud data”, *IEEE Trans. Parallel Distrib.*, vol. 25(1), pp. 222–233, 2014.
- [5] D. Cash, J. Jaeger, S. Jarecki, C.S. Jutla, H. Krawczyk, M. Rosu, M. Steiner, “Dynamic searchable encryption in very-large databases: Data structures and implementation”, *NDSS 2014*, USA, 2014.
- [6] D. Cash, S. Jarecki, C.S. Jutla, H. Krawczyk, M. Rosu, M. Steiner, “Highly-scalable searchable symmetric encryption with support for boolean queries,” *Advances in Cryptology- CRYPTO*

- [22] Z. Xia, X. Wang, X. Sun, Q. Wang, "A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data", *IEEE Trans. Parallel Distrib. Syst.* Vol. 27(2), pp. 340-352, 2016.
- [23] W. Zhang, Y. Lin, S. Xiao, J. Wu, S. Zhou, "Privacy Preserving Ranked Multi-Keyword Search for Multiple Data Owners in Cloud Computing", *IEEE Trans. Computers*, vol. 65(5), pp. 1566-1577, 2016.
- [24] X. Zhu, H. Dai, H. Yi, G. Yang, X. Li, "MUSE: An Efficient and Accurate Verifiable Privacy-Preserving Multi-keyword Text Search over Encrypted Cloud Data" *Security and Communication Networks* 2017, pp. 1-17, 2017.

اعظم سلیمانیان دارای مدرک



کارشناسی ارشد ریاضی از دانشگاه صنعتی امیرکبیر و دکترای ریاضی-رمز از دانشگاه خوارزمی تهران است. زمینه علمی مورد علاقه وی رمزگاری کاربردی است.
نشانی رایانامه ایشان عبارت است از:

soleimani1985@gmail.com

شهرام خزایی از سال ۱۳۹۱ استادیار



دانشکده علوم ریاضی در دانشگاه شریف است. وی مدرک کارشناسی و کارشناسی ارشد را در رشته مهندسی برق از دانشگاه صنعتی شریف (۱۳۷۷-۱۳۸۲) و مدرک دکترای علوم رایانه را از دانشگاه EPFL سوئیس (۱۳۸۹-۱۳۸۵) دریافت کرده است. زمینه‌های علمی-پژوهشی مورد علاقه وی علوم رایانه نظری، بهخصوص رمزگاری است.
نشانی رایانامه ایشان عبارت است از:

shahram.khazaei@sharif.ir

