

# بازیابی فازی داده‌های رمز شده با استفاده از داده‌ساختارهای چندمنظوره

اعظم سلیمانیان<sup>۱\*</sup> و شهرام خزایی<sup>۲</sup>

<sup>۱</sup>دانشگاه خوارزمی تهران، دانشکده ریاضی و علوم کامپیوتر، تهران، ایران

<sup>۲</sup>دانشگاه صنعتی شریف، دانشکده علوم ریاضی، تهران، ایران

## چکیده

با گسترش روزافزون سرویس‌های ابری، افراد حقیقی و حقوقی بیشتری تمایل به برون‌سپاری داده‌های خود روی این سرویس‌ها دارند؛ اما به دلایل امنیتی ترجیح می‌دهند قبل از برون‌سپاری داده آن را رمز کنند. رمز کردن 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 رمز شده 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 مرتبط‌تر می‌توان ترافیک شبکه را کاهش داد. داده‌ساختارها به عنوان بلوک‌های سازنده در رمزگذاری جستجوپذیر متقارن 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 اسناد در نظر گرفته شده است که با وجود کارایی بالا و سادگی، تا کنون در مبحث جستجو روی داده رمز شده 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.

واژگان کلیدی: رمزگذاری جستجوپذیر، جستجوی رتبه‌بندی شده، ساختمان داده، بازیابی فازی، پرسمان بولی.

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

Azam Soleimanian<sup>1\*</sup> & Shahram Khazaei<sup>2</sup>

<sup>1</sup>Department of Mathematics and Computer Science, Kharazmi University, Tehran, Iran

<sup>2</sup>Department 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.

داده ساختارهای 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 رتبه‌بندی‌شده این امکان را برای سرور فراهم می‌سازد تا نتایج را با توجه به میزان ارتباط آن‌ها به واژه مورد جستجو، به‌صورت رتبه‌بندی‌شده 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. به‌طور 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  $TF-IDF^3$  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 (زمانی‌که داده‌ها رمز شده هستند) به‌راحتی قابل اعمال نیست. در بازیابی فازی، با وجود عدم قطعیت در مورد حضور واژه در سند، جستجو همچنان امکان‌پذیر خواهد بود و همچنین با به‌کارگیری قوانین نظریه فازی، امکان اعمال کارای پرسمان‌های بولی<sup>4</sup> فراهم می‌شود.

## ۱- مقدمه

با گسترش خدمات ابری، مسأله تأمین امنیت داده‌ها در ابر به یکی از چالش‌های بزرگ در این حوزه تبدیل شده است. روش‌های رمزگذاری معمول جهت حفظ محرمانگی داده، منجر به ایجاد اختلال در برخی عملکردهای سرویس ابری، مانند عملکرد جستجو می‌شوند؛ از این‌رو، روش‌های رمزگذاری خاص‌منظوره، هم‌چون رمزگذاری جستجوپذیر، پا به عرصه ظهور نهاده‌اند که هم‌زمان با حفظ محرمانگی 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؛ به‌طوری‌که هر سند شامل تعدادی واژه است. مالک یک شاخص<sup>۱</sup> متناظر با اسناد خود تولید می‌کند که به‌طور 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 رمز شده و اسناد رمز شده را روی سرور ابر قرار می‌دهد. کاربری که تمایل به جستجو دارد، پرسمان خود را در قالب یک نشان<sup>۲</sup> (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.

رمز کردن شاخص یادشده به روش‌های مختلفی امکان‌پذیر است. به‌طور 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.

<sup>3</sup> Term frequency–inverse document frequency

<sup>4</sup> Boolean query

<sup>1</sup> Index

<sup>2</sup> Token

طرح ارائه شده در [8] از داده‌ساختاری استفاده می‌کند که متشکل از یک آرایه<sup>۳</sup>، یک فهرست پیوندی<sup>۴</sup> و یک جدول مراجعه<sup>۵</sup> است. این داده‌ساختار جستجوی تک‌واژه را در زمان بهینه (یعنی از مرتبه تعداد اسنادی که واژه را شامل می‌شوند)، فراهم می‌کند. آن‌ها همچنین داده‌ساختاری ارائه کردند که به قیمت افزایش تعداد نشان ارسالی از سمت کاربر، امنیت را افزایش می‌دهد. کش و همکارانش [6] داده‌ساختاری ارائه کردند 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 متنوع 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، 13، 9، 7، 5، 3] برای پشتیبانی از جستجوهای پویا، 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 متنوع روی پایگاه داده رابطه‌ای<sup>۶</sup> به‌ترتیب در [15، 2] و [12] یافت می‌شوند.\*

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

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 قابلیت‌هایی در داده‌ساختارهای یاد شده است؛ زیرا به‌سختی می‌توان داده‌ساختارهایی را پیدا کرد که اعمال تغییرات کوچک در آن‌ها، منجر به پشتیبانی کارا از عملکرد جدیدی شود. با توجه ویژه به چنین قابلیت‌هایی در داده‌ساختارهای مورد نظر و اعمال تغییرات ساده در آن‌ها، نوآوری‌های زیر را خواهیم داشت:

- بازیابی فازی را به‌عنوان روشی کارا، برای نخستین‌بار در جستجوی روی داده‌های رمز شده به‌کار می‌بریم.
- طرح رمز جستجوی رتبه‌بندی شده‌ای ارائه خواهیم کرد که قادر است در زمان بهینه، با دریافت پارامتر  $\alpha$  از سمت کاربر، کلیه اسنادی را که میزان ارتباط آن‌ها با واژه مورد جستجو دست‌کم به میزان  $\alpha$  است، شناسایی کند.
- طرح رمز جستجوی رتبه‌بندی شده‌ای ارائه می‌کنیم که قادر است در زمان بهینه، مرتبط‌ترین اسناد متناظر با یک پرسمان بولی را بازیابی کند.

## ۲- کارهای مرتبط

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

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

<sup>3</sup> Array

<sup>4</sup> Linked list

<sup>5</sup> Lookup table

<sup>6</sup> Dynamic

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

<sup>7</sup> Relational database

<sup>1</sup> Thorough scan

<sup>2</sup> Index-based

جستجوی رتبه‌بندی‌شده (با حضور چندین مالک داده) ارائه شده که تنها برای جستجوی تک‌واژه قابل اعمال است و از طرفی به بازیابی کل اسناد نیاز دارد. در این طرح همچنین از نگاشت دو خطی<sup>۱</sup> (جورساز<sup>۲</sup>) استفاده می‌شود که به شدت کارایی را کاهش می‌دهد. در [4]، روشی برای جستجوی رتبه‌بندی‌شده چندواژه‌ای (رابطه فصلی چندواژه) 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<sup>۳</sup> نیست. این ایده در [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 حاصل‌شده، مدت زمان جستجو همچنان به تعداد اسناد وابسته است. همچنین در [1]، روشی مبتنی بر رمزگذاری نیمه‌هم‌ریخت و معیار TF×IDF برای رابطه فصلی واژه‌ها ارائه شد که علاوه بر سرور یک پردازنده نیز در سیستم حضور دارد به طوری که فرض عدم تبانی سرور و پردازنده مطرح است. ابتدا با استفاده از رمزگذاری نیمه‌هم‌ریخت، سرور میزان ارتباط اسناد به پرسمان را به صورت رمز شده به دست می‌آورد؛ سپس مقادیر حاصل را به پردازنده ارسال و پردازنده با دسترسی به کلید رمزگشایی مقادیر را رمزگشایی و سپس دوباره رمزگذاری حافظه ترتیب را روی آن‌ها اعمال می‌کند و به سرور بر می‌گرداند. مشکل اساسی روش مطرح‌شده میزان پیچیدگی تعاملاتی بین سرور و پردازنده است که از مرتبه اندازه مجموعه اسنادی است که پرسمان فصلی را برآورده می‌کنند؛ به علاوه، پردازنده باید به ازای هر پرسمان عمل رمزگشایی مقادیر دریافتی از سرور و سپس عمل رمزگذاری حافظه ترتیب آن‌ها را انجام دهد؛ همچنین در [20]، از رمزگذاری تمام‌هم‌ریخت استفاده شده است که این گونه روش‌های رمزگذاری اغلب ناکارا هستند. در [16]، با استفاده از روش تسهیم راز به جای رمزگذاری نیمه‌هم‌ریخت، نیاز به رمزگذاری حافظه ترتیب از بین می‌رود؛ اما همچنان نیاز به بازیابی کل شناسه اسنادی که پرسمان را برآورده می‌کنند، 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

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 رمز نشده) است. معیار استفاده‌شده، با توجه به ویژگی‌های ذاتی‌اش، به نوع خود نیز موجب ایجاد انعطاف در طراحی داده‌ساختارها لازم و متعاقباً بهبود کارایی می‌شود. ما در این کار از داده‌ساختارهای مطرح‌شده در [8] و [6] به ترتیب به منظور افزایش کارایی جستجوی رتبه‌بندی‌شده 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 رتبه‌بندی‌شده 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 خواهند شد. در بخش ۴، شمای<sup>۴</sup> طرح رمز جستجوی رتبه‌بندی‌شده و مدل امنیت آن را ارائه خواهیم کرد. در بخش ۵، به بررسی داده‌ساختاری که کورتمولا و همکارانش [8] برای جستجوی تک‌واژه ارائه کرده‌اند، می‌پردازیم و نشان می‌دهیم که این داده‌ساختار قابلیت ترکیب با معیار بازیابی فازی و پشتیبانی از جستجوی رتبه‌بندی‌شده را دارد. در بخش ۶ داده‌ساختار ارائه‌شده 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 رتبه‌بندی‌شده 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  $q$  است؛ یعنی،  $\mathbb{Z}_q = \{0, \dots, 1\}$  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  $\{0, 1\}^*$  نشان می‌دهیم. نماد  $x||y$  بیان‌گر عملگر پیوست دو رشته  $x$  و  $y$  است. نماد  $\lambda$  را برای نمایش پارامتر امنیتی به کار می‌بریم. منظور از یک الگوریتم P.P.T، یک الگوریتم احتمالی با زمان اجرای چندجمله‌ای، بر حسب  $\lambda$  است. همچنین  $\text{poly}(\lambda)$  تابع چندجمله‌ای از  $\lambda$  را نمایش

<sup>1</sup> Bilinear map

<sup>2</sup> Paring

<sup>3</sup> Information retrieval

<sup>4</sup> Syntax

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

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

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

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

نمادهای روابط عطفی<sup>۵</sup>، فصلی<sup>۶</sup> و نقیض<sup>۷</sup> 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 به دنبال آن  $Ord(DB_F(q, \alpha))$  قابل تعریف هستند؛ به عنوان مثال برای پرسمان:

$$q = (w_1 \wedge w_2) \vee w_3$$

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

$$\begin{aligned} Enc(K, .): \mathcal{D} &\rightarrow \mathcal{R} \\ \text{باشد که برای } M, N \in \mathbb{N} \text{، دامنه و برد آن به ترتیب زیر است:} \\ \mathcal{D} = [M] &\stackrel{\Delta}{=} \{1, \dots, M\} \\ \mathcal{R} = [N] &\stackrel{\Delta}{=} \{1, \dots, N\} \end{aligned}$$

**تعریف ۱ (رمزگذاری حافظه ترتیب).** رمز متقارن یقینی  $OP\mathcal{E} = (Gen, Enc, Dec)$  را حافظه ترتیب گوئیم اگر برای هر  $i, j \in [M]$  و کلید  $K$  که توسط الگوریتم  $Gen$  تولید شده است، داشته باشیم:

$$i < j \Leftrightarrow Enc(K, i) < Enc(K, j)$$

امنیت رمزگذاری حافظه ترتیب به این معنی است که یک مهاجم کارا با دسترسی اوراکلی<sup>۸</sup> به تابع رمزگذاری و

می‌دهد. در این مقاله مدل مهاجم را صادق-اما-کنجکاو ( $HBC^1$ ) در نظر می‌گیریم. به این معنی که مهاجم الگوریتم‌ها را به درستی دنبال، اما سعی می‌کند مانند یک شنودگر، از طریق مشاهده داده‌های رمز شده، پرسمان‌های دریافتی و جواب‌های متناظرشان، اطلاعاتی را به دست آورد. یک تابع ناچیز<sup>۲</sup> از  $\lambda$  با نماد  $negl(\lambda)$  نشان داده می‌شود که از مرتبه  $\lambda^{-\omega(1)}$  است. به عبارت دیگر،  $negl(\lambda)$  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 درگیر شدن 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  $W_i$  مجموعه‌ای از چندین واژه و با شناسه  $id_i$  باشد که  $id_i \in \{0, 1\}^n$  و  $W_i \subseteq \{0, 1\}^*$  مجموعه  $W = \cup_{i=1}^d W_i$  بیان گر همه واژه‌های متمایز در مجموعه اسناد است که آن را واژه‌نامه می‌نامیم. پارامترهای  $d$  و  $n$  چند جمله‌ای از  $\lambda$  هستند. جهت سهولت، شناسه اسناد را اعداد طبیعی در نظر می‌گیریم، بنابراین، مجموعه:

$$ID = \{id \mid W_{id} \in DB\} = \{1, 2, \dots, d\}$$

بیان گر مجموعه همه شناسه‌ها است. همچنین،  $DB(w)$  نشان دهنده شناسه اسنادی است که واژه  $w$  را شامل می‌شوند؛ یعنی،  $DB(w) = \{id \mid w \in W_{id}\}$ . الگوی دسترسی<sup>۳</sup> به شناسه اسنادی گفته می‌شود که پرسمان را برآورده می‌کنند (به عنوان مثال برای تک واژه  $w$  الگوی دسترسی  $DB(w)$  است) و الگوی پرسمان<sup>۴</sup> 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:

$$F = \{ \{id, w, \mu_F(id, w)\} \mid id \in ID, w \in W \}$$

که  $\mu_F: ID \times W \rightarrow [0, 1]$  تابعی است که برای هر زوج مرتب  $(id, w) \in ID \times W$ ، اهمیت واژه  $w$  در سند با شناسه  $id$  را نشان می‌دهد و آن را تابع (درجه) عضویت می‌نامیم؛ همچنین تعریف می‌کنیم:

$$DB_F(w, \alpha) = \{ \{id, \mu_F(id, w)\} \} \quad (1)$$

<sup>5</sup> Conjunction

<sup>6</sup> Disjunction

<sup>7</sup> Negation

<sup>8</sup> Oracle access

<sup>1</sup> Honest-but-curious

<sup>2</sup> Negligible function

<sup>3</sup> Access pattern

<sup>4</sup> Query pattern

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

**تابع شبه تصادفی<sup>۱</sup>:** 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 تمایزناپذیر محاسباتی است. به طور رسمی، تعریف زیر را داریم:

**تعریف ۲ (تابع شبه-تصادفی).** فرض کنید  $\text{Func}[n, m]$  مجموعه کل توابع از  $\{0, 1\}^n$  به  $\{0, 1\}^m$  باشد. یک تابع  $F: \{0, 1\}^n \rightarrow \{0, 1\}^m$  را شبه تصادفی نامیم اگر در زمان چندجمله‌ای، روی ورودی داده شده، قابل محاسبه و همچنین برای هر مهاجم  $\mathcal{A}$  یک تابع ناچیز  $\text{negl}$  موجود باشد به طوری که:

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

یک جایگشت شبه تصادفی 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 رمزنگاری:** از جمله مسائل سخت ریاضی که در این مقاله از آن استفاده خواهیم کرد، فرض سختی مسأله دیفی هلمن تصمیمی است. به طور شهودی این فرض بیان می‌کند که اگر  $g^a$  و  $g^b$  داده شده باشند، آن گاه  $g^{ab}$  تصادفی به نظر می‌رسد. این مسأله به صورت زیر تعریف می‌شود:

**تعریف ۳ (فرض دیفی هلمن تصمیمی).**

گوییم مسأله DDH نسبت به  $\mathcal{G}$  سخت است اگر برای هر مهاجم  $\mathcal{A}$  یک تابع ناچیز  $\text{negl}$  موجود باشد به طوری که:

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

که  $a, b, c$  به طور یکنواخت از  $\mathbb{Z}_q - \{0\}$  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 رتبه بندی شده

یک پروتکل رمزگذاری جستجوپذیر متقارن رتبه بندی شده ( $\text{RSSE}^2$ )، یک پروتکل دو عاملی بین کاربر و سرور است. نحوه استفاده از این پروتکل به این صورت است که کاربر از

روی مجموعه اسناد خود، یک شاخص رمز شده تولید و آن را در سمت سرور ذخیره می‌کند. کاربر می‌تواند پرسمان‌های به فرم  $(q, \alpha)$  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 پرسمان  $q$ ، پارامتر  $\alpha$  را برآورده می‌کنند، به ترتیب نزولی اهمیت به دست آورد. پرسمان  $q$  می‌تواند تک واژه  $w$  یا یک عبارت بولی مانند  $w_1 \wedge w_2$  باشد. به ازای  $q = w$  خروجی پروتکل  $\text{Ord}(\text{DB}_F(w, \alpha))$  خواهد بود که  $\text{DB}_F(w, \alpha)$  در رابطه (۱) تعریف شده است.

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

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

▪  $V \leftarrow \text{Search}(K, q, \alpha; \text{EDB})$  بین کاربر و سرور اجرا می‌شود. کاربر کلید  $K$  و پرسمان  $(q, \alpha)$  و سرور شاخص رمز شده  $\text{EDB}$  را به عنوان ورودی‌های پروتکل تأمین می‌کنند. در پایان پروتکل، کاربر یک فهرست از شناسه اسناد 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، آن گاه پروتکل  $\text{Search}$  برای هر پرسمان، نتیجه درست را باز گرداند. این مفهوم در تعریف زیر دقیق شده است:

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

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

## ۴-۱- تعریف امنیت

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

<sup>3</sup> Adaptive-security

<sup>4</sup> Non-adaptive-security

<sup>1</sup> Pseudo-random function

<sup>2</sup> Ranked searchable symmetric encryption



را انتخاب می‌کند؛ یعنی،  $\mathcal{A}(1^\lambda) \leftarrow (\text{DB}; Q)$ . سپس بازی  $\mathcal{S}(\mathcal{L}(\text{DB}, Q))$  را اجرا می‌کند و خروجی آن را به مهاجم ارسال می‌کند؛ در نهایت  $\mathcal{A}$  یک بیت باز می‌گرداند که بازی از آن به عنوان خروجی خود، که با متغیر تصادفی  $\text{Simu}_{\mathcal{A}, \mathcal{S}}^\Sigma(\lambda)$  نشان داده می‌شود، استفاده می‌کند. گوییم طرح  $\Sigma$ ، دارای امنیت غیر تطبیقی با نشی  $\mathcal{L}$  است، اگر برای هر مهاجم  $\mathcal{A}$  یک شبیه ساز P.P.T مانند  $\mathcal{S}$  وجود داشته باشد، به طوری که:

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

## ۵- بازیابی فازی برای تک‌واژه

هدف این بخش، ارائه یک طرح رمز RSSE است که امکان بازیابی اسناد شامل تک‌واژه  $w$  را فراهم می‌آورد؛ به نحوی که میزان ارتباط اسناد به واژه مورد نظر حداقل  $\alpha$  باشد. این طرح که بر اساس [8] است، از داده ساختارهایی مثل آرایه، فهرست پیوندی و جدول مراجعه کمک می‌گیرد و تفاوت آن با [8] در مرتب چیدن اسناد و استفاده از رمزگذاری حافظ ترتیب است. همان‌طور که خواهیم دید، این تغییرات منجر به پشتیبانی از عملکرد مهم جستجوی رتبه‌بندی شده می‌شود. یکی از قابلیت‌های این داده ساختار امکان بازیابی اسناد 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 داده ساختار، توصیف قابلیت یاد شده و نحوه بهرمندی 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  $n$  عنصری  $L$  که در آرایه  $A$  ذخیره شده است، دنباله‌ای از عنصرهای  $N_i = (v_i, \text{addr}_A N_{i+1})$  است که  $1 \leq i \leq n$ ،  $v_i$  یک رشته دلخواه و  $\text{addr}_A(N_{i+1})$  نشانی عنصر بعدی فهرست در آرایه  $A$  است. تعداد عناصر فهرست  $L$  را به وسیله  $\#L$  نشان می‌دهیم. شاخص متشکل از دو داده ساختار زیر است (شکل (۱) را مشاهده کنید):

$A$ : یک آرایه که برای هر  $w \in W$ ، رمز شده  $\text{DB}_F(w)$  در آن ذخیره شده که  $\text{DB}_F(w)$  مطابق رابطه (۲) است.  
 $T$ : یک جدول مراجعه که برای هر  $w \in W$ ، اطلاعاتی برای بازیابی و رمزگشایی  $\text{DB}_F(w)$  از آرایه  $A$  فراهم می‌سازد.

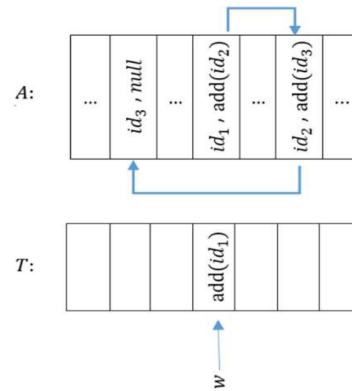
پرسمان را وابسته به پرسمان‌های ارسالی تا آن لحظه و پاسخ‌های دریافتی، انتخاب کند؛ اما در امنیت غیر تطبیقی مهاجم محدود به ارسال پرسمان‌ها به صورت یکجا است. به وضوح امنیت در برابر مهاجم تطبیقی، امنیت در برابر مهاجم غیر تطبیقی را نتیجه می‌دهد؛ اما عکس این مسأله صادق نیست. از طرف دیگر، طراحی کارای پروتکل‌های 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 رمز شده، هیچ اطلاعات مفیدی که به طور مستقیم منجر به آشکار شدن واژه تحت پرسمان یا شاخص شود، نشت پیدا نکند. کورتمولا و همکارانش [8] برای مدل کردن مفهوم عدم نشت اطلاعات مفید در یک طرح SSE، یک تعریف امنیت دقیق ارائه کردند؛ به این صورت که، با دریافت تنها یک نشی مشخص، یک شبیه ساز باید قادر باشد دید سرور را طی یک حمله غیر تطبیقی شبیه سازی کند. نشی یاد شده 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 اسناد است؛ بنابراین تضمین می‌شود که یک سرور HBC با داشتن شاخص رمز شده EDB و دریافت پرسمان‌های متفاوت، اطلاعاتی فراتر از آن چه که در نشی مشخص می‌شود، به دست نیاورد.

**تعریف ۶ (امنیت غیر تطبیقی [8]).** فرض کنید  $\Sigma = (\text{Setup}, \text{Search})$  یک طرح RSSE و تابع نشی  $\mathcal{L}$  یک الگوریتم حالت دار باشد؛ برای مهاجم  $\mathcal{A}$  و شبیه ساز  $\mathcal{S}$  بازی‌های زیر را تعریف می‌کنیم:

$\text{Real}_{\mathcal{A}}^\Sigma(\lambda)$ : مهاجم  $\mathcal{A}$  مجموعه اسناد DB و پرسمان‌های  $Q = \{(q_1, \alpha_1); \dots; (q_l, \alpha_l): l = \text{poly}(\lambda)\}$  را انتخاب می‌کند؛ یعنی،  $\mathcal{A}(1^\lambda) \leftarrow (\text{DB}; Q)$ . بازی الگوریتم  $\text{Setup}(\text{DB}) \leftarrow (K, \text{EDB})$  را اجرا و خروجی EDB را به مهاجم  $\mathcal{A}$  ارسال می‌کند. برای هر پرسمان  $(q_i, \alpha_i) \in Q$ ، بازی همچنین پروتکل  $V \leftarrow \text{Search}(K, q_i, \alpha_i; \text{EDB})$  را با ورودی  $(K, q_i, \alpha_i)$  برای کاربر و ورودی EDB برای سرور اجرا می‌کند و رونوشتی از تعاملات و خروجی کاربر را به  $\mathcal{A}$  می‌دهد؛ در نهایت  $\mathcal{A}$  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 اسناد DB و پرسمان‌های  $Q = \{(q_1, \alpha_1); \dots; (q_l, \alpha_l): l = \text{poly}(\lambda)\}$

به‌ازای هر  $w_i \in W$ ، یک فهرست پیوندی  $L_i$  تولید می‌کنیم که هر عنصر آن به‌ترتیب 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 به‌صورت رمزشده ذخیره می‌شوند. قسمت  $\mu_F(.,.)$  هر عنصر با یک روش رمزگذاری حافظ‌ترتیب و بخش دیگر با روش رمزگذاری معمول رمز می‌شوند. قبل از رمزکردن عنصر  $j$ -ام از فهرست  $L_i$ ، یک اشاره‌گر (نسبت به آرایه  $A$ ) به عنصر  $(j+1)$ -ام فهرست و یک کلید برای رمزکردن 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 نخستین عنصر هر فهرست در آرایه، از جدول مراجعه  $T$  استفاده می‌شود. هر درآیه در جدول  $T$  متناظر با یک  $w_i \in W$  است و یک زوج  $\langle address, value \rangle$  را شامل می‌شود. مؤلفه  $value$  محل و کلید رمزگشایی نخستین عنصر  $L_i$  در  $A$  را شامل می‌شود. خود  $value$  به‌وسیله یک تابع شبه‌تصادفی<sup>۱</sup> (PRF) رمز می‌شود. مؤلفه  $address$  برای تعیین محل درآیه‌های  $T$  به‌کار 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  $id_2$  و  $id_3$  بازیابی می‌شوند. توجه داریم که در این شکل، جهت سهولت، اطلاعات به‌صورت رمز نشده 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.

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  $\tilde{\alpha} = OPE.Enc(\alpha)$  به سرور ارسال می‌کند. سرور درآیه متناظر در  $T$  را پیدا و رمزگشایی می‌کند. در نتیجه، 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، به‌ترتیب بازیابی کند.

### ۵-۱- الگوریتم بازیابی فازی تک‌واژه (SKFR)

طرح رمز پیشنهادی (SKFR<sup>2</sup>) در الگوریتم زیر آورده شده که تفاوت‌های آن با طرح ارائه‌شده 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

##### الگوریتم Setup(DB):

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

##### ۱. ساخت آرایه $A$ :

به‌ازای هر  $w_i \in W$ ، فهرست  $L_i$  که شامل عناصر  $N_{ij}$  است به‌صورت زیر بساز و آن را داخل آرایه  $A$  ذخیره کن:

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

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

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

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

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

• اگر  $j = |DB_F(w_i)|$  قرار بده  $k_{ij} = \perp$  در غیر این صورت قرار بده  $k_{ij} \leftarrow SKE.Gen(1^{\lambda})$

• قرار بده

$$K_0 \leftarrow OPE.Gen(1^{\lambda})$$

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

و

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

<sup>2</sup> Single keyword fuzzy retrieval

<sup>1</sup> Pseudo-random function



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

اثبات: بازی شبیه سازی شده در تعریف ۶ به صورت زیر است:

۱. مهاجم  $\mathcal{A}$  مجموعه اسناد  $DB$  و پرسمان های  $Q = \{(w_1, \alpha_1); \dots; (w_l, \alpha_l): l = \text{Poly}(\lambda)\}$  را انتخاب می کند؛ یعنی،  $(DB; Q) \leftarrow \mathcal{A}(1^\lambda)$ .
۲. شبیه ساز  $\mathcal{S}$ ، به عنوان ورودی، اندازه واژه نامه، اندازه مجموعه اسناد (یعنی  $\sum_{w \in W} |DB(w)|$ ) و همچنین الگوی دسترسی  $\text{Ord}(DB_F(w_i, \alpha_i))$  و الگوی پرسمان را به ازای هر  $w_i$  دریافت می کند.
۳. شبیه ساز، اعداد تصادفی اما نزولی  $\mu_{ij}$  را انتخاب می کند و به ترتیب، به اعضای  $\text{Ord}(DB_F(w_i, \alpha_i))$  نسبت می دهد.

۴. به ازای هر پرسمان  $w_i$ ، شبیه ساز مشابه الگوریتم واقعی و با استفاده از ورودی های خود قادر است، سلول های متناظر  $A$  را شبیه سازی کند. در واقع، شبیه ساز  $\mathcal{S}$  در شبیه سازی آرایه  $A$ ، نیازی به  $w_i$  ندارد و داشتن فهرست  $\text{Ord}(DB_F(w_i, \alpha_i))$  به ازای واژه های موجود در پرسمان ها برای او کفایت می کند. شبیه ساز سایر سلول های خالی  $A$  را با مقادیر تصادفی پر می کند.
۵. به ازای هر پرسمان  $w_i$ ، شبیه ساز دو عدد تصادفی  $\gamma_i$  و  $\eta_i$  را به ترتیب از برد  $\pi$  و  $f$  انتخاب و مانند الگوریتم واقعی جدول  $T$  را شبیه سازی می کند (در واقع چون  $w_i$  را ندارد به جای  $\pi(K_3, w_i)$  و  $f(K_2, w_i)$  دو عدد تصادفی انتخاب می کند).

شبیه ساز سایر سلولهای خالی جدول  $T$  را با مقادیر تصادفی پر می کند.

۶. شبیه ساز داده ساختارهای  $A$  و  $T$  را خروجی می دهد. همچنین به ازای هر پرسمان  $w_i$ ، شبیه ساز مقدار  $t_i = \langle \gamma_i, \eta_i \rangle$  متناظر آن از مرحله قبل و همچنین عدد  $\tilde{\alpha}$  را خروجی می دهد به طوری که  $\tilde{\alpha}$  از تمام مقادیر  $\mu_{ij}$  در مرحله ۳، کوچک تر باشد.

با توجه به فرضیات قضیه، واضح است که بازی شبیه سازی 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  $\alpha$  به عبارت بولی:

- اگر  $|\text{DB}_F(w_i)| = j$  قرار بده  $\text{addr}_A(N_{i(j+1)}) = \text{NULL}$  در غیر این صورت:
- $\text{addr}_A(N_{i(j+1)}) = \phi(K_1, \text{ctr} + 1)$
- قرار بده  $N_{ij} = \langle v_{ij}, \text{addr}_A(N_{i(j+1)}) \rangle$
- عنصر  $N_{ij}$  را با کلید  $k_{i(j-1)}$  رمز کن و آن را درون آرایه  $A$  قرار بده به طوری که:
- $A[\phi(K_1, \text{ctr})] \leftarrow \text{SKF}. \text{Enc}(k_{i(j-1)}, N_{ij})$
- شمارنده  $\text{ctr}$  را یک واحد افزایش بده.

## ۲- ساخت جدول مراجعه $T$ :

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

## پروتکل $\text{Search}(K, w, \alpha, EDB)$ :

۱. کاربر: پیام  $t = \langle \pi(K_3, w), f(K_2, w) \rangle$  و  $\tilde{\alpha} = \text{OPE}. \text{Enc}(K_0, \alpha)$  را به سرور ارسال می کند.
۲. سرور: اعمال زیر را انجام می دهد؛
- \*  $t$  را به صورت  $\langle \gamma, \eta \rangle$  در نظر می گیرد و قرار می دهد  $\theta = T[\gamma]$
- \* اگر  $\theta \neq \perp$ ، آنگاه  $\theta \oplus \eta$  را به صورت  $\langle a_1, k_0 \rangle$  بازنویسی می کند، قرار می دهد  $j = 1$  و ادامه می دهد.
- در غیر این صورت نماد  $\perp$  را برمی گرداند.
- \* تا زمانی که  $a_j \neq \text{NULL}$  اعمال زیر را انجام می دهد؛
- $N_j = \text{SKF}. \text{Dec}(k_{j-1}, A[a_j])$  را به دست می آورد.
- $N_j$  را به شکل  $\langle v_j, a_{j+1} \rangle$  بازنویسی می کند 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 می کند. در غیر این صورت توقف می کند و ادامه نمی دهد.
- شمارنده  $j$  را یک واحد افزایش می دهد.

## ۲-۵- تحلیل امنیت الگوریتم SKFR

در این بخش به تحلیل امنیت الگوریتم پیشنهادی می پردازیم و نشان می دهیم این طرح دارای امنیت اثبات پذیر در مدل شبیه سازی است.

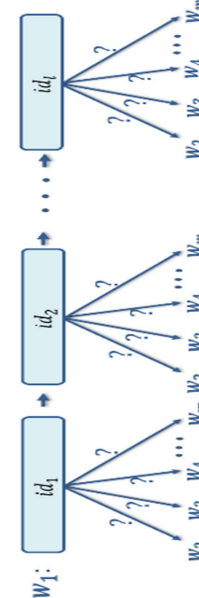
**قضیه ۱.** فرض کنید  $\text{OPE}$  یک طرح رمز حافظ ترتیب امن،  $\text{SKF}$  یک طرح رمز  $\text{CPA}^1$  امن،  $\phi$  و  $\pi$  دو  $\text{PRP}^2$  امن و  $f$  یک  $\text{PRF}$  امن باشد. آنگاه طرح رمز ارائه شده در الگوریتم

<sup>1</sup> Chosen-plaintext attack

<sup>2</sup> Pseudo-random permutation

$q = (w_1 \wedge w_2 \wedge \dots \wedge w_m)$  مربوط هستند (الگوریتم BooleanFR). فرض کنید واژه  $w_1$  با کمترین بسامد باشد، به این معنی که تعداد اسنادی که میزان ارتباط آن‌ها به واژه  $w_1$  حداقل  $\alpha$  است، کمترین تعداد باشد (در مقایسه با بقیه واژه‌های موجود در پرسمان  $q$ ).

برای جستجوی رتبه‌بندی‌شده پرسمان عطفی  $q$  در حالت رمز نشده، ابتدا شناسه اسنادی را که میزان ارتباط آن‌ها با واژه  $w_1$  حداقل  $\alpha$  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  $\mu_{ij}$  نشان می‌دهیم؛ در نهایت 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 رمز نشده.

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

شکل (۲) الگوریتم جستجو برای پرسمان عطفی را در حالت رمز نشده نشان می‌دهد؛ اما تعمیم این روش برای داده‌های رمز شده به‌طور بدیهی امکان‌پذیر نیست. یک تعمیم از این روش برای داده‌های رمز شده به‌صورت زیر است. ابتدا با استفاده از یک روش جستجوی رتبه‌بندی‌شده برای تک‌واژه (مانند الگوریتم SKFR) شناسه‌هایی که واژه  $w_1$  را شامل می‌شوند و دست‌کم به میزان  $\alpha$  با آن ارتباط دارند، در سمت سرور بازیابی می‌شوند. از طرف دیگر، واژه‌های فقط در اختیار کاربر هستند؛ لذا به‌منظور بازیابی  $\mu_{ij}$  متناظر با زوج

$(id_j, w_i)$  سرور و کاربر نیاز به همکاری و تعامل با یکدیگر دارند. بازیابی  $\mu_{ij}$  را می‌توان با استفاده از یک روش ابتدایی، اما ناکارا به‌صورت زیر انجام داد. سرور هرشناسه بازیابی‌شده  $id_j$  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 قبل در سمت سرور بارگذاری شده است و همچنین با استفاده از نشان دریافتی از طرف کاربر مقدار  $\mu_{ij}$  را از داده‌ساختار  $XSet$  بازیابی می‌کند. واضح است که این روش به‌طور کامل ناکارا است و نیاز به تعامل با کاربر دارد. درحالی‌که کاربردهای موجود در رایانش ابری همواره سعی دارند از تعامل با کاربر اجتناب شود. کش و همکارانش [6] روشی هوشمندانه را به‌منظور حذف این تعامل ارائه کردند. در روش آن‌ها درحقیقت  $XSet$  به‌عنوان یک مجموعه از عناصر تصادفی شبیه‌سازی می‌شود، به‌طوری‌که اگر  $id_j \in DB(w_i)$  آن‌گاه عنصر  $g^{Fp(K_I, id_j) \cdot Fp(K_X, w_i)}$  در  $XSet$  قرار داده می‌شود؛ یعنی با استفاده از توابع شبه‌تصادفی، عناصر موجود در  $XSet$  به‌نحوی ساخته می‌شود که شامل دوبخش هستند. یک بخش فقط شامل شناسه و بخش دیگر تنها شامل واژه است. بخشی که توسط کاربر تولید می‌شود و فقط شامل واژه است درحقیقت عضوی از یک گروه دوری است؛ به‌نحوی که مسأله دیفی‌هلمن تصمیمی در این گروه سخت است و لذا مقادیر متعدد تولیدشده 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 به‌طور ضمنی از روش کش و همکارانش بهره می‌برد. از آنجا که روش پیشنهادی کش و همکارانش جستجوی رتبه‌بندی‌شده 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؛ درحالی‌که 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  $XSet$  به‌دست آمده است؛ به‌نحوی‌که این داده‌ساختار در روش پیشنهادی ما دارای اندازه  $O(|ID| \times |W|)$  و زمان جستجوی  $O(1)$  است.

## ۱-۶- 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 ابتدا یک داده ساختار  $XSet$  را سمت سرور ذخیره می‌کند به طوری که متناظر با هر زوج  $(id_j, w_i) \in ID \times W$  یک سلول شامل مقدار  $\mu_{ij}$  موجود باشد. داده ساختارهای  $A$  و  $T$  نیز شبیه به داده ساختارهای مشابه در الگوریتم SKFR تولید و بر روی رسانه ذخیره سازی سرور قرار داده می‌شوند. جهت جستجوی پرسمان  $q = (w_1 \wedge w_2 \wedge \dots \wedge w_m)$  ابتدا مشابه روش قبل، کاربر درخواستی برای جستجوی واژه  $w_1$  به سرور ارسال و سرور اسناد با کمینه ارتباط  $\alpha$  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  $g^{F_p(K_i, id_j) \cdot F_p(K_X, w_i)}$  از داده ساختار  $XSet$  را به دست می‌آورند؛ به طوری که برای هر  $i = 2, \dots, m$  میزان ارتباط واژه  $w_i$  به سند  $id_j$  به صورت  $\mu_{ij}$  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:

$$m_j = \min\{\mu_{ij}; i = 1, \dots, m\}$$

در نهایت با در دست داشتن  $m_j$  ها، مرتبط ترین اسناد به عبارت بولی، به ترتیب به دست می‌آیند. نکته‌ای که باید مورد توجه قرار داد، این است که اسناد بازیابی شده، یعنی  $Ord(DB_F(w_1, \alpha))$ ، سمت سرور هستند و کاربر به آن‌ها دسترسی ندارد؛ بنابراین، برای محاسبه نشانی  $g^{F_p(K_i, id_j) \cdot F_p(K_X, w_i)}$  کاربر باید مقدار  $g^{F_p(K_X, w_i)}$  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 داده ساختار  $A$  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 پرسمان‌های قبلی، برای پرسمان جدیدی که سؤال نشده است، اسناد مربوطه را بازیابی کند. به عنوان مثال با استفاده از اطلاعات دریافتی برای پرسمان‌های دریافتی  $(w_1, w_2)$  و  $(w'_1, w'_2)$

<sup>۱</sup> توجه داریم که مقادیر  $g^{F_p(K_i, id_j) \cdot F_p(K_X, w_i)}$  در روش کش و همکاریانش به عنوان اعضای مجموعه  $XSet$  لحاظ شدند، ولی در روش ما به عنوان نشانی سلول‌های داده ساختار  $XSet$  هستند.

می‌تواند اسناد شامل پرسمان  $(w_1, w'_2)$  را بازیابی کند. برای رفع این مشکل، کش و همکاریانش پیشنهاد دادند که  $F_p(K_i, id_j)$  را به صورت کور شده سمت سرور ذخیره کنیم؛ یعنی به جای  $F_p(K_i, id_j)$  مقدار  $F_p(K_i, id_j) \cdot F_p(K_Z, w_1 || j)$  در سمت سرور قرار داده می‌شود و متعاقباً کاربر به جای  $g^{F_p(K_X, w_i)}$  عبارت  $g^{F_p(K_Z, w_1 || j) \cdot F_p(K_X, w_i)}$  را به سرور ارسال می‌کند (خواننده علاقمند می‌تواند به [6] مراجعه کند). در نهایت با ترکیب ایده ما و ایده موجود در [6]، الگوریتم مورد نظر ( $BooleanFR^2$ ) 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  $1 \leq j \leq |DB_F(w_i)|$  اعمال زیر را انجام بده:

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

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

• قرار بده

$$K_0 \leftarrow OPE.Gen(1^{\lambda})$$

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

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

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

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

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

و

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

• اگر  $j = |DB_F(w_i)|$  قرار بده  $addr_A(N_{i(j+1)}) = NULL$  در غیر این صورت:

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

• قرار بده  $N_{ij} = \langle v_{ij}, addr_A(N_{i(j+1)}) \rangle$

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

<sup>2</sup> Boolean fuzzy retrieval

• شمارنده  $ctr$  را یک واحد افزایش بده.  
 $A[\phi(K_1, ctr)] \leftarrow \mathcal{SKE}.Enc(k_{i(j-1)}, N_{ij})$

## ۲. ساخت جدول مراجعه $T$ :

به ازای هر  $w_i \in W$  قرار بده

۳. ساخت جدول مراجعه  $XSet$ :  
 $T[\pi(K_3, w_i)] = \langle \text{addr}_A(N_{i1}) || k_{i0} \rangle \oplus f(K_2, w_i)$

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

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

## ۴. خروجی:

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

## پروتکل $Search(K, q, \alpha, EDB)$ :

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

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

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

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

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

$xtoken[i, j] \leftarrow g^{Fp(K_Z, w_{i1} || j) \cdot Fp(K_X, w_i)}$

- 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  $\theta \neq \perp$ ، آن گاه  $\theta \oplus \eta$  را به صورت  $\langle a_1, k_0 \rangle$  بازنویسی

می‌کند، قرار می‌دهد  $V = \emptyset$ ،  $j = 1$  و ادامه می‌دهد.

در غیر این صورت نماد  $\perp$  را برمی‌گرداند.

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

•  $N_j = \mathcal{SKE}.Dec(k_{j-1}, A[a_j])$  را به دست می‌آورد.

•  $N_j$  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  $\tilde{\mu}_{1j} \geq \tilde{\alpha}$  آن گاه به ازای  $i = 2, \dots, m$  اعمال زیر

را انجام می‌دهد؛

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

می‌کند. سپس مقدار زیر را محاسبه می‌کند

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

- اگر  $m_j \geq \tilde{\alpha}$  آن گاه مجموعه  $V$  را به صورت  $V =$

$V \cup \{(\tilde{id}_{1j}, m_j)\}$  به روزرسانی می‌کند.

• در غیر این صورت پیام STOP را به کاربر ارسال می‌کند و همین جا متوقف می‌شود.

• شمارنده  $z$  را یک واحد افزایش می‌دهد.

\* مجموعه  $V$  را به طور نزولی بر حسب  $m_j$  ها مرتب می‌کند

و سپس این مجموعه مرتب را برمی‌گرداند.

مشاهده می‌شود که الگوریتم بالا به راحتی قابل تعمیم به هر

پرسمان بولی به صورت  $(w_1 \wedge \psi(w_2, \dots, w_m))$  است که

$\psi(\cdot)$  یک فرمولی بولی دلخواه است.

## ۲-۶- تحلیل امنیت الگوریتم BooleanFR

حال به تحلیل امنیت الگوریتم پیشنهادی خود می‌پردازیم. نشان می‌دهیم الگوریتم BooleanFR دارای امنیت بر مبنای شبیه سازی است.

۲. قضیه ۲. فرض کنید شرایط قضیه ۱ و همچنین  $F_p$  یک PRF امن و فرض DDH در گروه  $G$  برقرار باشد. آن گاه طرح رمز ارائه شده در الگوریتم های BooleanFR، دارای امنیت غیر تطبیقی با نشی  $\mathcal{L}$  است (که نشی  $\mathcal{L}$  در طول اثبات به عنوان ورودی شبیه ساز مشخص شده است).

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

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

$Q = \{(q_1, \alpha_1); \dots; (q_l, \alpha_l) : l = \text{poly}(\lambda)\}$

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

۲. واضح است که شبیه ساز  $\mathcal{S}$ ، به عنوان ورودی اندازه واژه نامه، اندازه مجموعه اسناد (یعنی

$|\sum_{w \in W} |DB(w)||$ )، همچنین  $|ID| \times W$  و

$|Ord(DB_F(w_1, \alpha))|$ ، اندازه الگوی دسترسی یعنی

$|Ord(DB_F(q, \alpha))|$  و الگوی پرسمان را به ازای مؤلفه

نخست پرسمان (یعنی  $w_1$ )، دریافت می‌کند. فرض

کنیم مهاجم دو پرسمان 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

$id \in Ord(DB_F(w_1, \alpha)) \cap$

$Ord(DB_F(w'_1, \alpha'))$ .

در این صورت نشی که شبیه ساز به عنوان ورودی

دریافت می‌کند؛ به صورت زیر است:

شبیه ساز می‌داند دومین مؤلفه در  $q$  با سومین مؤلفه

در  $q'$  مساوی است. همچنین، اطلاعاتی در مورد

با توجه به فرضیات قضیه، بازی شبیه سازی بالا از بازی واقعی تمایزناپذیر است.

## ۷- نتیجه گیری

در این مقاله نشان داده شد که با استفاده از قابلیت هایی که داده ساختارها فراهم می کنند، می توان عملکرد طرح های موجود را افزایش داد. از جمله عملکرد جستجوی رتبه بندی شده مورد بررسی قرار گرفت. بازیابی فازی به عنوان روشی کارا در حوزه بازیابی اطلاعات برای جستجوی رتبه بندی شده روی داده های رمز شده به کار برده شد. در جدول (۱) روش های پیشنهادی خود (SKFR) و (BooleanFR) را با روش های مشابه (جستجوپذیر رتبه بندی شده برای تکواژه یا عبارت بولی) مورد مقایسه قرار داده ایم. همان طور که در این جدول مشاهده می شود، طرح ما، برای پشتیبانی از پرسمان های تکواژه ای، نسبت به طرح های [21] و [23] به طور قابل توجهی کارا تر است؛ همچنین، مقایسه طرح پیشنهادی ما با طرح های مشابه [4] و [22]، برای پشتیبانی از پرسمان های بولی، کارایی آن نسبت به طرح های یاد شده را نشان می دهد.

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

$\text{Ord}(\text{DB}_F(w_1, \alpha)) \cap \text{Ord}(\text{DB}_F(w'_1, \alpha'))$  می آورد از جمله ناتهی بودن آن [6].

۳. شبیه سازی داده ساختارهای  $A$  و  $T$ ، به ازای مقادیر مشترک در دو طرح الگوریتم SKFR و BooleanFR، مشابه قضیه ۱ انجام می شود. به ازای مقادیر جدیدی که در این طرح نسبت به طرح الگوریتم SKFR داریم، یعنی مقادیر  $z_{ij}$ ،  $\tilde{d}_{ij}$  و  $xid_{ij}$  مقادیر تصادفی به ترتیب از برد  $F_p$ ،  $\text{SKF} \cdot \text{Enc}'$  و  $F_p$  انتخاب می شوند.

۴. شبیه سازی داده ساختار  $XSet$ : ابتدا شبیه ساز به ازای هر  $w \in W$  مقدار تصادفی  $f$  از برد  $F_p$  را انتخاب می کند (در واقع چون شبیه ساز  $w$  را ندارد به جای  $f_p(K_X, w)$  مقدار تصادفی انتخاب می کند).

حال به ازای هر پرسمان  $q = w_1 \wedge \dots \wedge w_m$  سلول های متناظر به صورت زیر شبیه سازی می شوند: به ازای هر واژه  $w_i$  که  $i = 2, \dots, m$ ، مقادیر  $f_i$  مذکور در بالا استخراج می شوند.

شبیه ساز همچنین مقادیر  $xid$  را (متناظر با  $w_1$  و اعضای  $\text{Ord}(\text{DB}_F(w_1))$  که در مرحله ۳ محاسبه کرده است، استخراج و سلول های مورد نظر را مطابق الگوریتم واقعی می سازد؛ به طوریکه مقادیر  $\mu_{ij}$  مورد نیاز به طور تصادفی انتخاب شده اند. در پایان سلول های خالی با مقادیر تصادفی پر می شوند.

۵. شبیه سازی مقادیر  $xtoken[i, j]$ : می دانیم در الگوریتم واقعی داریم:

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

شبیه ساز به جای مقادیر

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

مقادیر متناظر را که در مراحل ۲ و ۳ محاسبه کرده است، در نظر می گیرد و  $xtoken[j]$  را مطابق الگوریتم واقعی محاسبه می کند.

۶. شبیه سازی نشان  $t$  و مقدار  $\tilde{\alpha}$  مانند قضیه ۱ انجام می شود؛ به طوری که  $\tilde{\mu}_{1j} \geq \tilde{\alpha}$  به ازای:

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

و همچنین، به تعداد  $|\text{Ord}(\text{DB}_F(q, \alpha))|$  از این اندیس های  $j$  در نامسای  $\tilde{\alpha} \geq m_j$  صدق می کنند که  $m_j$  مطابق الگوریتم واقعی محاسبه می شود.

درواقع مقادیر  $\mu_{ij}$  طوری انتخاب می شوند که شرایط بالا روی  $\tilde{\alpha}$  امکان پذیر باشند.

۷. شبیه ساز داده ساختارهای  $A$ ،  $T$  و  $XSet$  و همچنین نشان  $t$ ،  $\tilde{\alpha}$  و  $xtoken[j]$  را خروجی می دهد.

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. K p  , 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.

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

(جدول-۱): مقایسه با کارهای مشابه  
(Table-1): Comparison with related works

در این جدول،  $|W|$  نشان دهنده اندازه واژه‌نامه،  $d$  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  $\alpha$  کمیت مورد نظر کاربر برای ارتباط سند به واژه است و  $N$  اندازه مجموعه اسناد است، یعنی،  $N = \sum_{w \in W} DB(w)$

## 8- References

## ۸- 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 رمزنگاری 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 (۱۳۷۷-۱۳۸۳) و مدرک دکترای علوم رایانه را از دانشگاه EPFL سوئیس (۱۳۸۹-۱۳۸۵) دریافت کرده است. زمینه‌های علمی-پژوهشی مورد علاقه وی علوم رایانه نظری، به خصوص رمزنگاری است. نشانی رایانامه ایشان عبارت است از:

shahram.khazaei@sharif.ir

