Continuous Top-K Monitoring on Document Streams

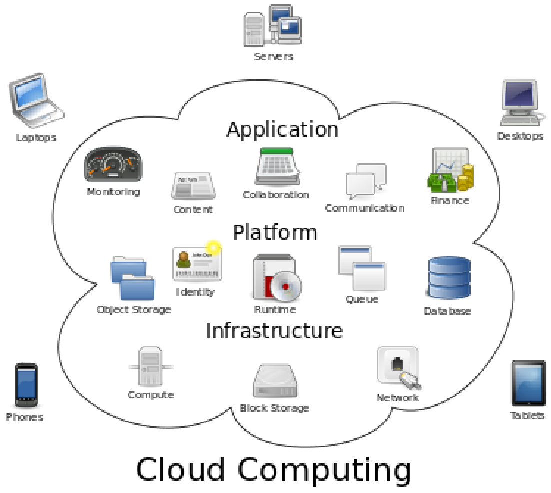
Ms.R.Vinnarasi[1], Mr. E. Ranjith[2]
[1, 2] Dept of MCA
[1, 2] Krishnasamy College of Engineering and Technology

Abstract- *The querying and analysis of data streams has been a topic of much recent interest, motivated by applications from the fields of networking, web usage analysis, sensor instrumentation, telecommunications, and others. Many of these applications involve monitoring answers to continuous queries over data streams produced at physically distributed locations, and most previous approaches require streams to be transmitted to a single location for centralized processing. Unfortunately, the continual transmission of a large number of rapid data streams to a central location can be impractical or expensive. We study a useful class of queries that continuously report the k largest values obtained from distributed data streams ("top-k monitoring queries"), which are of particular interest because they can be used to reduce the overhead incurred while running other types of monitoring queries. We show that transmitting entire data streams is unnecessary to support these queries and present an alternative approach that reduces communication significantly. In our approach, arithmetic constraints are maintained at remote stream sources to ensure that the most recently provided topk answer remains valid to within a user-specified error tolerance. Distributed communication is only necessary on occasion, when constraints are violated, and we show empirically through extensive simulation on real-world data that our approach reduces overall communication cost by an order of magnitude compared with alternatives that offer the same error guarantees.*

Keywords- Top-k query; Continuous query; Document stream.

I. INTRODUCTION

The instance, in securities exchange, a constant top-k inquiry (top-k question for short) can be utilized to screen ongoing exchanges and henceforth recover the 10 most noteworthy exchanges inside the most recent 30 minutes.
The inquiry results could assist financial specialists with tracking market hotspots and settle on reasonable choices.



II. LITERATURE REVIEW

A literature review is an account of what has been published on a topic by accredited scholars and researchers. Occasionally you will be asked to write one as a separate assignment, but more often it is part of the introduction to an essay, research report, or thesis. In writing the literature review, your purpose is to convey to your reader what knowledge and ideas have been established on a topic, and what their strengths and weaknesses.

III. PROPOSED METHODOLOGY

The Proposed Authorized Keyword Search

An attribute-based keyword search with efficient user revocation scheme for keyword space W and access structure space G consists of nine fundamental algorithms as follows:

- Setup (λ, N) → (PK, MK): The setup algorithm takes as input the security parameter λ and an attribute universe description N. It defines a bilinear group G of prime order p with a generator g. Thus, a bilinear map is defined as e: $G \times G \rightarrow G1$, which has the properties of bilinear, computability and non-degeneracy. It outputs the public parameters PK and the master secret key MK. The version number ver is initialized as 1.

- CreateUL(PK, ID) → UL: The user list generation algorithm takes as input PK and the user identity ID. It outputs the user list UL for a dataset.
- EncIndex(PK, GT, w) → D: The index encryption algorithm takes as input the current PK, the access structure GT ∈ G, a keyword w ∈ W and outputs the encrypted index D.
- KeyGen (PK, MK, S) → SK: The key generation algorithm takes as input the current PK, the current MK, and the attribute set S associated with a particular user. It outputs the user's secret key SK.
- ReKeyGen (Φ, MK) → (rk, MK′, PK′): The re-encryption key generation algorithm takes as input the attribute set Φ, and the current MK. It outputs a set of proxy re-encryption keys rk for all the attributes in N, the updated MK′ and PK′, where all the version numbers are increased by 1. For the attributes not in Φ, set their proxy re-encryption keys as 1 in rk.
- ReEncIndex(Δ, rk, D) → D′: It takes as input an index D, rk and the attribute set Δ. Then it outputs a new re-encrypted index D′.
- ReKey (Ω, rk, PSK) → PSK′: It takes as input a user's partial secret key PSK, rk and the attribute set Ω. Finally, it outputs a new PSK′ for that user.
- GenTrapdoor(PK, SK, w′) → Q: The trapdoor generation algorithm takes as input the current PK, the user's SK, a keyword of interest w′∈ W and outputs the trapdoor Q for the keyword w′.
- Search(UL, D, Q) → search result or ⊥: The search algorithm takes as input the user list UL, the index D and the user's trapdoor Q. It outputs valid search result or returns a search failure indicator ⊥

IV. MODULE DESCRIPTION

1. Similarity Measure:

We treat the query (i.e., the set of keywords it specifies) and the documents as vectors. Letting T be the dictionary of all terms, a query or a document vector includes one weight per term in the dictionary.

2. Document Freshness:

In stream monitoring applications, the freshness of information is essential. Hence, a focusing on the most recent stream documents is required. The two prevalent formulations to achieve this focusing are the *sliding window* and the *decay model*. A sliding window only considers as valid the documents that arrived most recently; the sliding window includes either a fixed number of documents (count-based version) or those that arrived within a fixed number of time units before current time (time-based). On the other hand, in the decay model the score of the documents drops over time by applying a decay function, so that the more recent documents are favored in query answering.

3. Index & Query Processing for Snapshot Queries

In this section, we overview the ID-ordering paradigm for snapshot queries. The documents are indexed by an inverted file, comprising a list Li for every term ti in the dictionary. Li holds an entry hdID; fii for every document that includes term ti (where dID is the ID of the document, and fi its weight for term ti). All lists are sorted in ascending document ID. The execution strategy to process a (snapshot) query **q** on this index evaluates the documents one after another from the sorted lists, but it performs "jumps" over zones of document IDs.

4. Reverse ID-Ordering for CTQD processing

A straightforward approach is to index documents as per normal, and evaluate all CTQDs. Whenever a new document arrives, we need to (i) update the index and (ii) reevaluate each and every query. This approach is impractical because it requires excessive processing for index maintenance and, primarily, for query reevaluation from scratch. A key observation to remedy the problem is that an arriving document may affect the result of just a fraction of the queries.

V. FUTURE SCOPE

In future, at extending the approach to reduce the number of trapdoors under multi owners setting and also to provide the solution for KASE in the case of federated clouds.

VI. CONCLUSION

Considering the practical problem of privacy preserving data sharing system based on public cloud storage which requires a data owner to distribute a large number of keys to users to enable them to access his/her documents, we for the first time propose the concept of key-aggregate searchable encryption (KASE) and construct a concrete KASE scheme. Both analysis and evaluation results confirm that our work can provide an effective solution to building practical data sharing system based on public cloud storage. In a KASE scheme, the owner only needs to distribute a single key to a user when sharing lots of documents with the user, and the user only needs to submit a single trapdoor when he queries over all documents shared by the same owner. However,if a user wants to query over documents shared by multiple owners, he must generate multiple trapdoors to the cloud. How to reduce the number of trapdoors under multi-owners setting is a future work. Moreover, federated clouds have attracted a lot of attention nowadays, but our KASE cannot be applied in this case directly. It is also a future work to provide the solution for KASE in the case of federated clouds.

REFERENCES

[1] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting Your Right: Attribute-based Keyword Search with Fine-grained Owner- enforced Search Authorization in the Cloud," in IEEE INFOCOM, pp. 226-234, 2014.

[2] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable,and fine-grained data access control in cloud computing," in Proc.of IEEE INFOCOM, pp. 1-9, 2010.

[3] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," IEEE TPDS, vol. 24, no. 1, pp. 131-143, 2013.

[4] S. Kamara and K. Lauter, "Cryptographic cloud storage," in Finan-cial Cryptography and Data Security, pp. 136–149, 2010.

[5] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. of IEEE S&P, pp. 44-55, 2000.

[6] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in USENIX Security Symposium, vol. 201, no. 1, 2011.

[7] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dis-sertation, Stanford University, 2009.