Confidentiality-Preserving Rank-Ordered Search

A. Swaminathan,[†] Y. Mao,[†] G.-M. Su,[†] H. Gou,[†] A. Varna,[†] S. He,[†] M. Wu,[†] and D. Oard[‡]
[†]Department of Electrical & Computer Engineering and [‡]College of Information Studies
Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742
{ashwins, ymao, gmsu, hmgou, varna, shanhe, minwu, oard}@umd.edu

ABSTRACT

This paper introduces a new framework for confidentiality preserving rank-ordered search and retrieval over large document collections. The proposed framework not only protects document/query confidentiality against an outside intruder, but also prevents an untrusted data center from learning information about the query and the document collection. We present practical techniques for proper integration of relevance scoring methods and cryptographic techniques, such as order preserving encryption, to protect data collections and indices and provide efficient and accurate search capabilities to securely rank-order documents in response to a query. Experimental results on the W3C collection show that these techniques have comparable performance to conventional search systems designed for non-encrypted data in terms of search accuracy. The proposed methods thus form the first steps to bring together advanced information retrieval and secure search capabilities for a wide range of applications including managing data in government and business operations, enabling scholarly study of sensitive data, and facilitating the document discovery process in litigation.

Categories and Subject Descriptors: H.3.3 [Information Systems]: Information Search and Retrieval

General Terms: Algorithms, Design, Security

Keywords: Secure index, encrypted domain search, ranked retrieval

1. INTRODUCTION

In the current information era, efficient and effective search capabilities for digital collections has become essential for information management and knowledge discovery. Meanwhile, a growing number of collections are professionally maintained in data centers and stored in encrypted form to limit their access to only authorized users in order to protect confidentiality and privacy. Examples include medical records, corporate proprietary communications, and sensitive government documents. An emerging critical issue that must be addressed is how to protect data collections and indices through encryption, while providing efficient and effective search capabilities to authorized users.

Cryptographic encryption protects data from compromise due to theft or intrusion. In addition to outsider attacks, security measures should also be taken against potential insider attacks. For example, when information storage is outsourced to a third-party data center, system administrators and other personnel involved may not be trusted to have decryption keys and access the content of the data collections. When an authorized user remotely accesses the data collection to search and retrieve desired documents, the large size of the collections often makes it infeasible to ship all encrypted data to the user's side, and then perform decryption and search on the user's trusted computers. Therefore, new techniques are needed to encrypt and organize the data collections in such a way as to allow the data center to perform efficient search in encrypted domain.

There are a number of scenarios where the content owner may want to grant a user limited access to search a confidential collection. For example, the searcher could be a scholar or a low-level analyst who wants to identify relevant documents from a private/classified collection, and may need clearance only for the top-ranked documents; the searcher could also be the opposing side during document discovery phase of a litigation, who would request relevant documents from the content owner's digital collection (say, emails) be turned over.

The requirements of balancing privacy and confidentiality with efficiency and accuracy pose significant challenges to the design of search schemes for a number of search scenarios. This problem has attracted interests from the cryptography community in recent years to investigate theories and techniques for "searchable encryption." However, existing work only supports Boolean searches to identify the presence/absence of terms of interests in encrypted documents. Advances in information retrieval have gone well beyond Boolean searches; scoring schemes have been widely employed to quantify and rank-order the relevance of a document to a set of query terms [1]. The goals of this paper are to explore a framework to securely rank-order documents in response to a query, and develop techniques to extract the most relevant document(s) from a large encrypted data collection. To our best knowledge, this is the first attempt in the research community to explore secure rank-ordered search. As an initial step, we focus in this paper on modeling common scenarios of secure rank-ordered search and exploring indexing and search techniques built upon existing

StorageSS'07, October 29, 2007, Alexandria, Virginia, USA.
Copyright 2007 ACM 978-1-59593-891-6/07/0010...$5.00.

established cryptographic primitives. The understandings obtained from this exploration will pave ways to bring together researchers from information retrieval [1] and applied cryptography [2] to establish a bridge between these areas.

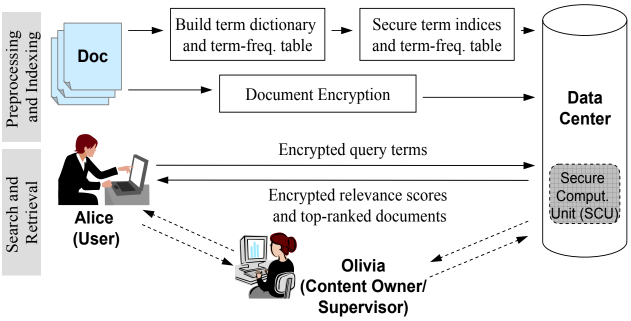
To accomplish our goals, we collect term frequency information for each document in the collection to build indices, as in traditional retrieval systems for plaintext. We further secure these indices that would otherwise reveal important statistical information about the collection to protect against statistical attacks. During the search process, the query terms are encrypted to prevent the exposure of information to the data center and other intruders, and to confine the searching entity to only make queries within an authorized scope. Utilizing term frequencies and other document information, we apply cryptographic techniques such as order-preserving encryption to develop schemes that can securely compute relevance scores for each document, identify the most relevant documents, and reserve the right to screen and release the full content of relevant documents. The proposed framework has comparable performance to conventional searching systems designed for non-encrypted data in terms of search accuracy.

The rest of this paper is organized as follows. Related background and prior work are reviewed in Section 2. Section 3 discusses representative use scenarios and Section 4 introduces a baseline model for supporting secure and efficient search. We then discuss in Section 5 the use of order-preserving encryption to allow data center to compute relevance scores while still maintaining confidentiality. Experimental results are presented in Section 6, and conclusions are drawn in Section 7.

2. BACKGROUND AND PRIOR ART

There has been a considerable amount of prior work on algorithms and data structures to support information retrieval for plaintext documents focussing on various issues, including efficient representation [1] and effective ranking [3]. In contrast, protection of sensitive information in the document collection, the indices, and/or the queries has received much less attention until recently. Some exploration of search in encrypted data and private information retrieval systems has been reported in [4, 5, 6]. These techniques generally involve high computational complexity in search, or incur a considerable increase in storage to store specially encrypted documents. Approaches to reduce search complexity were introduced in [7, 8], at an expense of limited search capabilities confined by a keyword list identified beforehand. The documents containing some of the pre-identified keywords are first found, and the keywords or the keyword indices are encrypted in a way that facilitates search and retrieval. These existing techniques target simple Boolean searches to identify the presence or absence of a term in an encrypted text. Much of the existing work has not been applied to large collections, and it is not clear whether it can be easily extended to more sophisticated relevance-ranked searches.

To facilitate the development of secure rank-ordered search, we briefly review the concept of term frequency statistics of a collection, which are widely used for ranked retrieval of unencrypted documents. Consider a data collection that contains $N^{(D)}$ documents, in which $N^{(T)}$ unique terms appear. The term frequency (TF) information for all terms and all documents can be organized as a table of size $N^{(T)} \times N^{(D)}$, in which the entry at i^{th} row and j^{th} column indicates the number of occurrences of the i^{th} term in the j^{th} document. The TF is then employed to define the relevance score for rank-ordering documents in a collection. One example is the Okapi [3] relevance score $CW(i,j)$, which is defined as:



Preprocessing and Indexing
Doc
Build term dictionary and term-freq. table
Secure term indices and term-freq. table
Document Encryption
Data Center
Search and Retrieval
Encrypted query terms
Encrypted relevance scores and top-ranked documents
Secure Comput. Unit (SCU)
Alice (User)
Olivia (Content Owner/ Supervisor)

Figure 1: A framework for confidentiality-preserving ranked search.

$$CW(i,j) = \frac{CFW(i)\ TF(i,j)\ (K_1+1)}{K_1(1-b+b\cdot NDL(j)) + TF(i,j)}. \quad (1)$$

Here $NDL(j) = L(j)/L_{avg}$ represents the normalized length of the j^{th} document and is obtained by dividing the length of the j^{th} document, $L(j)$, by the average document length L_{avg}; $N(i)$ is the number of documents containing the i^{th} term; $CFW(i)$ denotes the collection frequency weight of the i^{th} word:

$$CFW(i) = \log(N^{(D)}/N(i)); \quad (2)$$

where K_1 and b are constants chosen to achieve the best retrieval effectiveness for the particular collection. Example values are $K_1 = 2$ and $b = 0.75$. Given a query consisting of a single term $w(i)$, the set of relevance scores $\{CW(i,j), j = 1,\ldots,N^{(D)}\}$ can be directly used to identify the most relevant documents. If a query contains multiple terms $\{w(i_1), w(i_2), \ldots, w(i_M)\}$, the relevance scores for each of the query terms are added, *i.e.* $\sum_{i_k=i_1}^{i_M} CW(i_k,j)$, $\forall j$, and this overall score is employed to rank-order the documents.

3. SCENARIOS FOR SECURE SEARCH

This section presents several representative scenarios where the secure search over a document collection may take place. As shown in Fig. 1, the content owner, Olivia, uses the services of a data center to store a large number of documents, as well as perform search and retrieval. Olivia may also grant another user Alice the permission to search and retrieve her documents through the data center. In this case, we refer to Olivia as the supervisor. In addition, to prevent leakage of information against potential hackers' break-in, the documents stored at the data center are encrypted. The supervisor manages the content decryption keys and may provide decryption services upon Alice's request. In the following, we examine a few application scenarios under this framework.

- *Case 1:* The content owner, Olivia, wants to search for some documents stored at the data center. She has a limited bandwidth connection with the data center, and needs to search through the encrypted content without downloading the entire collection; she does not trust data center with her unencrypted content; she wants to remotely search and retrieve top-ranked relevant documents without revealing the search terms, document content, and document index information to the data center. We refer to this scenario as *confidentiality preserving baseline model* and will discuss it in more detail in Section 4.

• *Case 2:* Now, consider the scenario where a user Alice, who is not the content owner, wants to search for a particular phrase in the set of confidential documents held by the data center. This scenario could arise in situations such as analyzing a private/classified collection, or recovering facts in a litigation process. In general, Olivia does not trust the data center with the document content or the term frequency values. However, we consider that the data center has a secure computing unit (SCU), which is trusted by Olivia to some degree. Depending on the level of trust on the SCU by the content owner, we identify the following scenarios:

− *Case 2a:* Olivia trusts the SCU both with the plaintext documents and the associated term-frequency table.

− *Case 2b:* Olivia trusts the SCU with the plaintext term-frequency values, but not with the plaintext documents.

− *Case 2c:* Olivia does not trust SCU with either the term-frequency values or the documents in plaintext form, but trusts SCU with certain computations to be performed on some encrypted version of the term-frequency table without disclosing the exact values.

In *Case 2a* and *Case 2b*, Olivia trusts SCU with the term frequency values. In this case, the SCU can be considered as a heavily guarded “Maximum-Security Computing Unit” (MaxSCU) in the data center that can be used to decrypt TF table, compute relevance scores using (1), and rank-order the documents based on these values. The baseline model we introduce in Section 4 can be a solution to this scenario. The MaxSCU, however, is a critical link of the overall system security and may therefore be subject to attack. As such, it can be expensive to design and maintain such a unit hosted in a data center. In *Case 2c*, an adversary’s threat of breaking SCU is alleviated as the SCU only sees an encrypted version of the term-frequency index and not the plaintext values. This scenario calls for two layers of encryption to allow the SCU to compute relevance scores in the encrypted-domain of the first layer and to enhance confidentiality outside SCU with an outer-layer encryption. In Section 5, we will present a scheme to accomplish this objective. If Olivia does not trust the SCU with any plaintext or encrypted data, Olivia’s involvement would be required for computing the relevance score. Thus it would reduce to the baseline model discussed in the next section.

4. CONFIDENTIALITY-PRESERVING BASELINE MODEL

In this section, we develop a framework to perform ranked search securely and efficiently with minimum disclosure of the indexing information. We assume that the data center can only be trusted with data storage and should not be allowed to obtain information about the stored data. The framework consists of two major stages, a *pre-processing stage* for building a secure term frequency table and a secure inverse document frequency table, and a *search stage* for rank-ordering documents in response to a particular query while preserving the confidentiality of TF information. We first develop a *baseline* model that involves multiple rounds of interaction between the client and server to obtain the relevant information pertaining to a query; we will improve upon this baseline model in later sections.

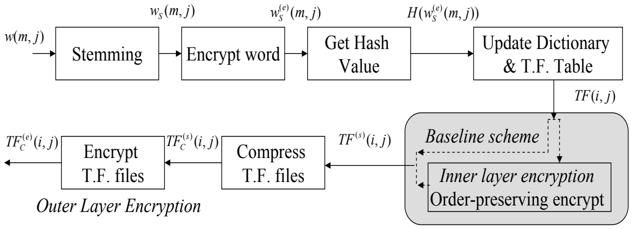


Figure 2: Generating and securing index information.

4.1 Indexing and Generating Secure Term Frequencies

The pre-processing is executed once by Olivia, when she stores the documents, all in encrypted form, in the data center. The major task of the pre-processing stage is to build a secure term frequency table and a secure inverse document frequency table, so as to facilitate efficient and accurate information retrieval.

For an unprotected term frequency table, both the search term and its term frequency information are in plaintext. To protect the confidentiality of the search, we encrypt each of them in an appropriate way. As shown in Fig. 2, a word w in a document first undergoes stemming to retain the word stem and to remove the word ending. The stemmed word w_S is then encrypted using an encryption function E and the *word-key* K_{w_S}- to obtain the encrypted word $w_S^{(e)} = E(K_{w_S}, w_S)$. Here the word-key is unique to each stemmed word and is obtained with a key derivation function. $w_S^{(e)}$ is further mapped to a particular row i in the term frequency table, where the index i is established via a hashing function such that $i = H(w_S^{(e)})$. The term frequency information is collected by counting the number of occurrences of the stemmed word in the j^{th} document, and stored in the table entry $\{TF(i,j)\}$. This process is repeated to obtain the term frequencies for all terms and documents, and the TF values are then further encrypted.

In the *baseline* model, the data center is only trusted with storing data. There is a single layer of encryption to protect the term frequency information from both unauthorized users and from the data center. We first encode each row of the term frequency table to minimize the required storage. The encoded term frequency table denoted by TF_C is then encrypted to create $TF_C^{(e)}$, as $TF_C^{(e)}(i,.) = E(K_i^{(TF)}, TF_C(i,.))$, where a key $K_i^{(TF)}$ is used to encrypt the i^{th} row of the term frequency table $TF_C(i,.)$. To increase security, the value of $K_i^{(TF)}$ is unique for each row and is derived from the word-key K_{w_S} corresponding to the i^{th} row. Thus, compromising the key corresponding to one row does not compromiseing other rows of the term frequency table.

Since computing the relevance score requires the use of collection frequency weight (CFW) of a word as in (1), the CFW can be computed before-hand and encrypted using the same word key as in the term frequency table. The CFW

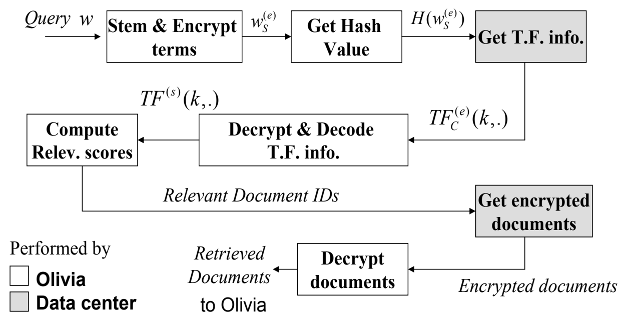


Figure 3: Search and Retrieval in the baseline model

is then stored in the data center separately from the term frequency. It can be sent to Olivia along with the term frequency rows for computing relevance scores.

Given the sparsity of the TF table, proper encoding the term frequency rows helps reduce the bandwidth required for its transmission during the search phase. In our work, we use *value-precision* encoding to compress the term-frequency rows, wherein we encode the position and the value of every non-zero term in the term-frequency table. Our results with 200,000 emails from the *Enron* email corpus [9] suggest that the average size of the compressed term frequency rows is 435 bytes, which reduces the size of uncompressed term frequency table by 1/460; 86% of the term frequency rows have size within 200 to 300 bytes. Thus, the encoding helps keep the required bandwidth to transmit the term frequency rows modest.

4.2 Rank-Ordered Search

In the baseline model, search and retrieval is initiated by the content owner, Olivia. As shown in Fig. 3, when searching for a particular word w in the collection, Olivia first performs stemming to obtain the stemmed word w_S. The word-key is then derived from the master key and used to encrypt the stemmed-word w_S to obtain $w_S^{(e)}$. After that, the hash value of $w_S^{(e)}$ is calculated and sent to data center. Using the received hash value $k = H(w_S^{(e)})$, the data center searches the protected term frequency table $TF_C^{(e)}$ and identifies the row corresponding to the query word w. In this way, we conceal the query content from the data center.

After the data center identifies the target row $TF_C^{(e)}(k, .)$ from the encrypted term frequency table $TF_C^{(e)}$, that particular row $TF_C^{(e)}(k, .)$ is sent back to Olivia, who then decrypts and decodes to obtain the plaintext term frequencies $\{TF(k, j), \forall j\}$. Olivia further computes relevance scores from the term frequency values according to (1), rank-orders the documents based on the score, and requests the most relevant documents from the data center. When a query consists of multiple terms, $w(i_1)$, $w(i_2)$, ..., $w(i_M)$, these M corresponding rows in TF table are identified, and sent back to Olivia for computing relevance scores. Olivia uses the received information to compute the relevance scores for each term, and then combines them to obtain the final scores.

4.3 Discussion

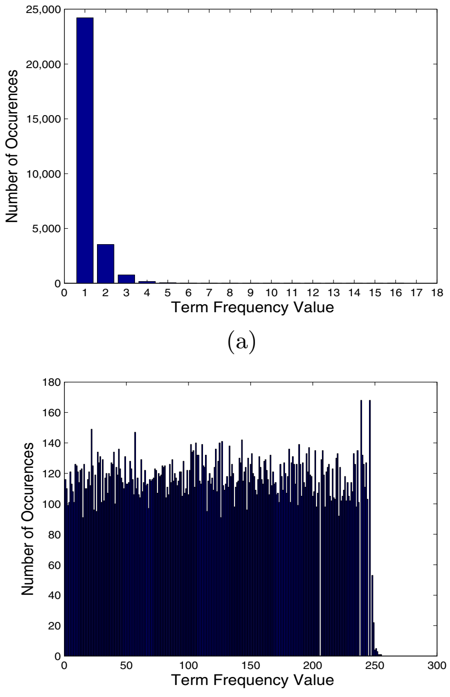
In the baseline model, the data center does not get access to the unencrypted content at any point of time both during the pre-processing and the search and retrieval stage. It does not know the TF information, as they are stored encrypted. The only information that the data center gains from the search process is the retrieval log. The retrieval log would at most contain data on which user searched for what encrypted queries, when and how often. The data center could also learn which documents were requested pertaining to the encrypted search queries. Based on such information collected over a period of time, the data center might perform statistical attacks. However, such attacks can be mitigated by the content owner, Olivia, through adding to her requests some phantom terms and phantom document indices to diffuse the access statistics of her intended terms and documents. Olivia can also hide her identity by introducing a proxy in her connection link with the data center.

5. SECURE RANKING OF RELEVANCE

The baseline model introduced in the previous section addresses the scenarios where the content owner makes a query himself/herself. In this section, we present an alternate scheme to enable a search capability from a user other than the content owner. This scheme reduces the involvement of Olivia by shifting the task of computing the relevance score to the data center, while still maintaining the confidentiality of the term-frequency information and the document content. To remove the need for communications between the data center and content owner during content search, we must be able to perform computations and ranking directly on term-frequency data in its encrypted form. We refer to this searchable layer of encryption as the *inner-layer encryption*, which is denoted by $TF^{(s)}$. Inner-layer encryption can be done via cryptographic tools such as homomorphic encryption (HME) and order preserving encryption (OPE); the computation of relevance score should be adapted accordingly to support encrypted domain computation. We use OPE in this paper to demonstrate the concept for secure ranking of relevance. After the inner-layer encryption, $TF^{(s)}$ is encoded to obtain $TF_C^{(s)}$, and further encrypted to obtain $TF_C^{(e)}$ in the same way as in the baseline scheme. We refer to this second round of encryption as *outer-layer encryption*, which prevents unauthorized users from accessing TF information.

The indexing and pre-processing stages of the proposed schemes are similar to the baseline model with an additional inner-layer encryption. When searching for a particular query consisting of multiple terms, $w(i_1)$, $w(i_2)$, ..., $w(i_M)$, in the collection, Alice first performs stemming and sends the stemmed words to the content owner, Olivia, who checks whether Alice has the required permission to search for the query words. Upon verification, Olivia derives the word-keys from the master key and uses it to encrypt the stemmed-words to obtain $w_S(i_k)^{(e)}, k = 1, 2, \ldots, M$. The hash value of $w_S(i_k)^{(e)}$ is then calculated and transmitted to Alice who forwards it to the data center. Using the received hash values $H(w_S(i_k)^{(e)})$, the data center searches the protected term frequency table $TF_C^{(e)}$ and identifies the rows corresponding to the query words, without obtaining plaintext information about the query.

After the data center identifies the target rows from the term frequency table $TF_C^{(e)}$, it uses the SCU to decrypt and decode it, and subsequently obtain the corresponding rows of the term frequency table $TF^{(s)}$ that are protected by the



(a)

(b)

Figure 4: (a) TF distribution from Enron email corpus, and (b) TF distribution after order preserving encryption.

inner-layer encryption algorithms. During this stage, the encrypted rows, $TF^{(s)}$, are retained within the SCU and not revealed to the data center. The SCU then performs the entire computations for the relevance scores directly in the encrypted domain, rank-orders the documents, and sends back the most relevant document identifiers with their ranking.

The order preserving encryption (OPE) operation on $TF(i,j)$ to obtain encrypted $TF^{(s)}(i,j)$ is designed such that if $TF(i,j) < TF(i,k)$, then $TF^{(s)}(i,j) < TF^{(s)}(i,k)$. Due to the monotonicity of the relevance score function in (1), as long as the order of relevance scores (or the order of term frequency values) is preserved, the correct search results can be obtained for queries that involve only one term. However, such an OPE approach employing one-to-one mapping [10] cannot be directly employed to secure the TF values. In Fig. 4(a), we show the histogram of the TF values over all the words from the dictionary built using the *Enron* email corpus [9]. As can be seen in the figure, the TF histogram is very peaky and therefore one-to-one OPE mapping would not be able to randomize such TF values retaining the peaky nature, which might result in information leak to the server. In order to enhance security and reduce the amount of leak in term-frequency information, an appropriate one-to-many mapping is desired to flatten the peaky distribution to be close to a uniform distribution and increase its randomness.

Using the one-to-many OPE method, we encrypt each row of the TF table corresponding to each of the $N^{(TF)}$ terms. The peaky structure of term frequency distribution reflects that there are a large number of entries having the same term frequency value. In order to flatten the peaky distribution, we map every entry $TF(i,j)$ with the value tf to a random number in the range of $[tf^l, tf^u]$, where $0 \leq tf^l \leq tf^u < 2^B$ are the lower bound and the upper bound of the random mapping range ($B = 8$ in our experiment). In order to make the one-to-many mapping an order preserving operation, for two adjacent term frequency values tf_1 and tf_2, their random mapping ranges $[tf_1^l, tf_1^u]$ and $[tf_2^l, tf_2^u]$ are chosen to be non-overlapping but close to each other, *i.e.*, if $tf_1 \lesssim tf_2$, then $tf_1^u \lesssim tf_2^l$. To maximize the entropy of the encrypted output, the random mapping range $[tf^l, tf^u]$ for a term frequency value tf is adaptively determined according to the distribution of raw TF values, so that an approximately uniform distribution can be obtained for the encrypted values $TF^{(s)}(i,j)$. Our algorithm chooses the size of the random mapping range $[tf^l, tf^u]$ proportional to the histogram of the values of tf in that particular row. The values of tf^l and tf^u are then determined with the above constraints. In this way, an approximately uniform distribution can be obtained for the encrypted $TF^{(s)}(i,j)$ at individual rows of the TF table.

Applying the proposed random mapping method to the actual histogram in Fig. 4(a), with the random mapping range individually determined for each row, we obtain encrypted $TF^{(s)}(i,j)$ with the histogram shown in Fig. 4(b). We can see that we indeed obtain approximately uniform distributions after the one-to-many order preserving encryption, even though the distributions of raw term frequency values are quite different in these two examples. This suggests that the disclosure of term frequency information to unauthorized users and the data center that carries out the search task can be minimized.

By introducing the order-preserving encryption on raw term frequency values, the OPE enables document search on the data center side while preventing it from learning the critical term frequency information. When a query contains a single term, the OPE can achieve effective search as the baseline model by accurately identifying the target documents. As the number of terms in a query increases, the order may not be completely preserved when summing up scores of all terms. To examine the search accuracy for multiple terms, we compute the differences in Mean Average Precision (MAP) for the baseline scheme and for the order-preserving encryption scheme for different numbers of search terms. Our results show that with multiple terms in a query, the accuracy of OPE is within a small gap of around 0.06 from that of the baseline model. As shall be shown in Table 1, the precision at 10 documents (P@10) values show no noticeable reduction in the expected number of relevant documents on the first page of a typical results display (from 0.49 to 0.46).

6. RESULTS AND DISCUSSIONS

In this section, we compare the performance of the baseline model and OPE in terms of security, retrieval accuracy, and examine the tradeoffs involved in securing the term frequency using order preserving encryption. We evaluate the retrieval accuracies of the secure search schemes on the W3C collection, with the 59 queries used for the discussion search in the enterprise track in the 2005 Text Retrieval Conference (TREC) [12]. Any document that is judged partially relevant or relevant is taken to be relevant in our test (i.e. conflating the top two judgement levels). We study the per-

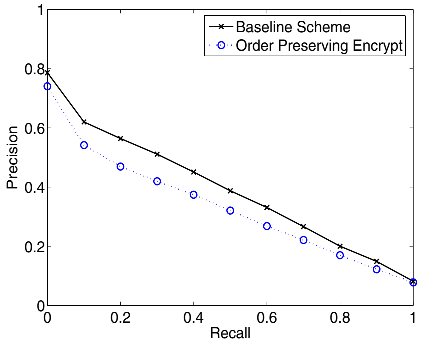


Figure 5: Precision-recall graphs for two schemes.

formance of these schemes using precision-recall graphs. The precision-recall results for all 59 queries are collected and the average is shown in Fig. 5. We notice from the figure that the retrieval accuracy of the OPE is slightly lower than that of the baseline scheme. However, this small drop in performance in OPE comes with added advantages of fewer communication rounds between the user and the server for search and retrieval than the baseline schemes.

We also examine the retrieval accuracy of the proposed schemes using a set of common evaluation metrics discussed in the literature [11, 12]. The evaluation results are shown in Table 1. Comparing with the results published in [12] with the values in Table 1, we find that the baseline scheme using the Okapi relevance score would have been ranked *second* in the evaluation, suggesting that the retrieval accuracy for our baseline scheme is about as good as the state of the art in the information retrieval literature. With regards to the OPE, we notice that even with the added layer of security, the performance would have appeared in the *top five* retrieval schemes evaluated in the TREC 2005 conference.

The promising results of the proposed framework also suggest tradeoffs among security, storage efficiency, search accuracy, and system complexity. As efficient storage of term frequency is needed, our present inner layer encryption in OPE retains the sparsity of the TF table by leaving zero-valued terms unchanged rather than encoding (at least some of) them. In this case, the SCU may gain knowledge of the zero-valued TF, but does not know for which plaintext term and which document. Our proposed schemes also presently require a secure environment to initially generate the encrypted indices and encrypted documents. Usually such initial processing is required only once. However, in the case when the collection is constantly changing, the secure index information in OPE should also be updated. In particular for the OPE scheme, the mapping of frequency values for all terms that appear in the new/ changed documents should be updated to best balance security and search accuracy. Our future work will investigate how to enable incremental changes to the encrypted TF.

7. CONCLUSIONS

In this work, we develop a framework for confidentiality-preserving rank-ordered search in large scale document collections. We explore techniques to securely rank-order the documents and extract the most relevant document(s) from an encrypted collection based on the encrypted search queries. We present several representative scenarios depending on the security requirement; and develop techniques to perform efficient search and retrieval in each case. The proposed method maintains the confidentiality of the query as well as the content of retrieved documents.

Table 1: Retrieval accuracy measures for various schemes.

Metric	Baseline	OPE	Metric	Baseline	OPE
MAP	0.3739	0.3142	P@20	0.4271	0.3839
r-prec	0.3878	0.3476	P@30	0.3791	0.3271
bpref	0.3798	0.3412	P@100	0.2366	0.2056
P@5	0.5424	0.5017	P@1000	0.0471	0.0422
P@10	0.4881	0.4627	RR1	0.7257	0.6749

The techniques introduced in this work are first attempts to bring together advanced information retrieval capabilities and secure search capabilities. In addition to our focus on securing indices, other important security issues include protecting communication links and combating traffic analysis. These will need to be addressed in future work. Further investigations of complete cryptographic modeling, efficient algorithm design, and system evaluations can shine light on an improved balance between the security, efficiency, and accuracy of search, leading to a wide range of applications, such as searching information with hierarchical access control, and flexible "e-discovery" practices for digital records in legal proceedings.

8. REFERENCES

[1] I.H. Witten, A. Moffett, and T.C. Bell. *Managing Gigabytes*, Morgan Kaufmann, 2nd ed., 1999.

[2] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions," *Proc. of the ACM Comp. and Comm. Security (CCS)*, Oct. 2006.

[3] S. E. Robertson and K. S. Jones, "Simple Proven Approaches to Text Retrieval," *Technical Report TR356*, Cambridge Univ. Computer Laboratory, 1997.

[4] R. Brinkman, J. M. Doumen, and W. Jonker, "Using Secret Sharing for Searching in Encrypted Data," *Workshop on Secure Data Management in a Connected World*, LNCS 3178, pp. 18-27, Aug. 2004.

[5] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private Information Retrieval," *J. ACM*, vol. 45, no. 6, pp. 965–982, 1998.

[6] D. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," *IEEE Sym. on Research in Security and Privacy*, pp. 44-55, May 2000.

[7] D. Boneh, G. Crescenzo, R. Ostrovsky, G. Persiano, "Public-key Encryption with Keyword Search," *Proceedings of Eurocrypt*, 2004.

[8] E-J. Goh, "Secure Indexes," *Cryptology ePrint Archive, Report 2003/216*, 2003.

[9] B. Klimt and Y. Yang, "Introducing the Enron Corpus," *Conf. on Email and Anti-Spam (CEAS)*, Mountain View, CA, 2004.

[10] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order Preserving Encryption for Numeric Data," *Proc. of SIGMOD*, Jun. 2004.

[11] D. K. Harman, "Common Evaluation Measures," *in Appendix, Proceedings of Text Retrieval Conference*, 2005. Available online at `http://trec.nist.gov/`

[12] N. Craswell, A. P. de Vries, and I. Soboroff, "Overview of the TREC-2005 Enterprise Track," *Text Retrieval Conference*, 2005.