Abstract: Protection of content of sensitive text documents is important in enterprise intranets. An index structure is needed to support efficient search and retrieval, but it can lead to information leakage; by statistical attacks an adversary can draw probabilistic inference about the contents of document collection. Zerr and others present a confidential index structure and the ranking of retrieved documents for the query, but only for single-term queries. The solution proposed in the paper generalizes Zerr’s method by using an anonymization parameter and query-dependent anonymized inverse document frequency factors; thereby it provides better ranking and gives possibility of multi-term queries.

Key-Words: information retrieval, confidential text indexing, inverted index, posting list, inverse document frequency, r-confidentiality, anonymization, ranking

1 Introduction

Number of sensitive documents shared over enterprise intranets is growing rapidly. Sharing of access-controlled documents has traditionally been accomplished through centralized repositories such as shared file directories on intranet servers, access-controlled web pages, wikis, and even source code control systems. There are two ways to accomplish it. The first type of the solutions allows to users to publish their own documents on access-controlled web pages on their own web server, whose administrator they trust. In case of the second type the documents can be placed on a public server after encrypting it with a key known only to the dedicated persons. In both types, just securing the documents is insufficient, as they need to be indexed to support efficient search and retrieval [12].

Inverted indexes are the standard choice for search of documents by keywords. An inverted index is a sequence of items containing information about occurrences of a term in a document, where terms (generalization of words) can be found in the gathered dictionary of the document set (corpus). Each item in the list – which records that a term appeared in a document – is conventionally called a posting list element or simply posting. The list is then called posting list. Fig. 1 shows an inverted index with three posting lists and eight posting list elements. The index contains sufficient information to reconstruct the set of words in a sensitive document, so it needs to be protected against unauthorized access.

We focus on problem of search for sensitive unstructured documents shared within collaboration groups. These groups work in a large enterprise or in multiple enterprises, the groups are established and are closing down relatively quickly, as projects start and finish. A person may be a member of more groups, but he/she can belong to a limited number of collaboration groups only. Document owners create and distribute new documents, so the content of the shared documents changes over time, as does the group membership. Usually there is no central authority that all members trust with the content of their documents; however the group members are willing to trust the enterprise’s authentication facilities. Such environments are common in large companies, large universities, and large government groups as well.

![Example for inverted index with three posting lists](image-url)
It is unlikely that all users can agree on a single trusted central authority to enforce access control on index entries; even if they can, centralized indexes are attractive targets for attack and will need additional protection. For example, even if the exact content of the elements is disguised, the details of posting lists may disclose confident information. Protecting an inverted index is a challenging problem when there is no single trusted central authority to enforce access control on posting list elements.

There are several privacy-preserving techniques to provide protection of sensitive data [1], including k-anonymity solutions [6][9], data mining methods [11], encryption [3][8]. The idea in k-anonymity is to reduce the granularity of representation of the data in such a way that a given record cannot be distinguished from at least (k−1) other records. Privacy-preserving data mining methods can include perturbation, blocking, merging, swapping or sampling tools for data modification. Perturbation is accomplished by the alteration of an attribute value by a new value (i.e. changing a 1-value to a 0-value, or adding noise). Blocking means the replacement of an existing attribute value with a fix sign or character representing the missing value. Merging is the combination of several values into a coarser category. Data swapping refers to interchanging values of individual records [1]. Sampling refers to releasing data for only a sample of a population. These tools can be well used for continuous variables or qualitative variables, but the protection requirements for string variables and inverted index are totally differ. There are some implemented frameworks [3][8] for protection of data by encryption (e.g. XML data [4]). Although encryption is a standard technique for storing data confidentially, however it is not possible to directly apply this technique to secure an inverted index. Even if posting list elements are encrypted, they can leak critical statistical data. Therefore dedicated methods are required for protecting inverted index structures against attacks aiming at statistical information on the original content of corpus. There are solutions by Zerr and others [12][13], which take the so-called statistical attacks into the consideration, but their search abilities are strongly restricted: users cannot use more than one keyword in the query and the ranking of answers based on only limited information for the highest anonymity. We will present an extended concept to improve these restrictions by introducing an anonymity parameter, while it defends against the statistical attacks.

2 Confidentiality of indexing
To set up the potential dangers, we consider the typical goals of an adversary’s statistical attack on an index:

• Attack 1: Adversary tries to reconstruct the exact content or the term frequencies (TF) – i.e. the number of occurrences of each term in the document – of an access-restricted document. Exact reconstruction is clearly undesirable, but the term frequency distribution is sufficient to characterize the subject of a long document, and the likely content of a short email.

• Attack 2: Adversary tries to determine the document frequencies (DF) for the set of access-restricted documents at a participating site. The document frequency is the number of documents that contain a particular term. In an ordinary inverted index, the length of a term’s posting list is its document frequency. There is an information leak of these frequency distributions about the owner’s global activity. For example, even if the exact content of the elements is obscured, the lengths of the posting lists can tell an industrial spy which compounds are used in the development of a new chemical process [5], or can call attention for a novel solution of an enterprise.

• Attack 3: Adversary tries to determine whether a particular term appears in a particular access-restricted document at a particular site, or at any indexed site. For example, an employee wants to know whether his colleague, Jack Mamoulian has participated in a confidential project. Therefore rare terms like “Mamoulian” need special protection in order to restrict the access to content of project documents.

An adversary will already have some background knowledge about the possible contents of a document collection. The aim is to restrict his ability to increase this knowledge, even if he can examine the content of the index server. Let us investigate a scenario where an adversary tries to reconstruct collection content from an inverted index. From his background knowledge B he will know a-priori that a term t is contained in document d with an estimated probability P(t is in d). The estimated probability P(X|I,B) about fact X based on B cannot be controlled, but his ability to refine that estimate when he computes P(X|I,B) can be limited,
where \( I \) is the index structure that he can access. In the following, only facts \( X \) of “term \( t \) is in document \( d \)” and “term \( t \) is not in document \( d \)” will be considered.

The concept of \( r \)-confidentiality was introduced in [12] to measure the degree of information that can leak from an index about access-restricted documents, given an adversary’s background knowledge of the document corpus (or general language statistics). An indexing scheme is \( r \)-confidential iff \( P(X|I, B) \leq r \cdot P(X|B) \). The \( r \)-confidentiality focuses on confidentiality of document content. In addition, secure communication channels such as https should be also required to provide confidentiality for the content of queries and answers, but we do not consider this issue here.

Here, \( r \) is the factor of maximal probability amplification for term \( t \) in \( d \) at given \( I \). The indexing scheme offers maximal protection when \( P(X|B) = P(X|I, B) \), i.e. \( I \) does not provide any additional knowledge about \( X \). The \( r \)-confidentiality restricts the ability of an adversary to draw probabilistic inference about the contents of a document collection.

### 3 Zerber \( r \)-confidential indexing

There is a simple implementation of \( r \)-confidential inverted indexing for sensitive documents, called Zerber [12], which relies on a centralized set of largely untrusted index servers that hold posting list elements encrypted with Shamir secret sharing scheme (\( k \) out of \( n \) secret sharing) [10]. The dictionary of Zerber’s inverted index scheme contains all of the terms in the corpus and a pointer at each term to beginning of the corresponding postings list. Each postings list stores the list of documents in which a term occurs and term frequencies. To prevent the adversary from learning a term \( t \)’s document frequencies, the \( t \)’s posting list is combined with the posting lists for several other terms, as shown in Fig. 2 for an unencrypted posting list. A merged posting list element hence contains three fields: \([\text{document\_ID}, \text{term\_ID}, \text{TF}]\), where document\_ID is identity of document, term\_ID is identity of term and TF is the term frequency. Naturally the relationships between a document and its identity number as well as a term and its identity number are disguised.

![Fig. 2 Merged posting list](image)

Although the adversary cannot determine the exact term in an element, he can draw probabilistic inference about it using his background knowledge, e.g. general language statistics. Let \( n_d(t_i) \) is the number of documents in corpus D containing term \( t_i \). Supposed that all posting elements of terms have been merged into one list, this list may contain \( n_L \) posting elements:

\[
   n_L = \sum_{t_i \in D} n_d(t_i) \tag{1}
\]

The probability of the posting element containing a particular term \( t_u \), is the ratio of number of posting element with term \( t_u \) and \( n_L \), that is \( p_u = n_d(t_u) / n_L \). The conditional probability of the posting element containing a particular term \( t_u \), provided that it is one of the terms in the set \( S = \{t_1, \ldots, t_n\} \), is the ratio of number of posting element with term \( t_u \) and the length of posting list built from set \( S \), i.e.:

\[
   n_d(t_u) / n_S, \text{ where } n_S = \sum_{t_i \in S} n_d(t_i) \tag{2}
\]

For an inverted indexing scheme to be \( r \)-confidential, this probability should not exceed \( r \) times the probability of \( t_u \) occurring in a document according to the adversary’s background knowledge:

\[
   \sum_{t_i \in S} n_d(t_i) \leq r \cdot \frac{n_d(t_u)}{n_L} \tag{3}
\]

Therefore the requirement for \( r \)-confidentiality can be derived as [12]:

<table>
<thead>
<tr>
<th>Terms:</th>
<th>Posting list elements:</th>
</tr>
</thead>
<tbody>
<tr>
<td>aim</td>
<td>DocA.pdf, 15</td>
</tr>
<tr>
<td>t_1</td>
<td>d_3</td>
</tr>
<tr>
<td>project</td>
<td>C.pdf, 45</td>
</tr>
<tr>
<td>t_2</td>
<td>d_4</td>
</tr>
<tr>
<td></td>
<td>B.doc, 4</td>
</tr>
<tr>
<td></td>
<td>d_2</td>
</tr>
<tr>
<td></td>
<td>DocA.pdf, 24</td>
</tr>
<tr>
<td></td>
<td>d_5</td>
</tr>
</tbody>
</table>

Merged posting list:

\[
\begin{array}{c}
\text{List}_1 \\
\end{array}
\begin{array}{c}
\text{d}_3, \text{t}_1, 15 \\
\text{d}_3, \text{t}_2, 24 \\
\text{d}_2, \text{t}_1, 7 \\
\text{d}_2, \text{t}_2, 4 \\
\end{array}
\]

Naturally the relationships between a document and its identity number as well as a term and its identity number are disguised.
There are heuristic merging strategies that satisfy this formula and minimize the expected query cost by trading off between the degree of confidentiality \( r \) and the index size \( n_L \). The Depth First Merging (DFM) strategy [12] tries to minimize \( r \) under a predetermined maximum value for \( M \), the number of merged posting lists. For a given \( r \)-value the Breadth First Merging (BFM) strategy [12] sorts terms into posting lists by minimizing \( M \).

Some remarks for the usage of the Zerber indexing scheme and procedure are the followings:

- The server authenticates the user and determines the groups the user belongs to. For this purpose, each index server records which users belong to each group, and which posting elements are accessible to each group.
- To index a document, the document is divided into terms; an algorithm encrypts them and gives each one an ID that will be globally unique within its posting list. Based on these the merged posting lists will be updated.
- To execute a query, the user first authenticates himself, the index servers determine his groups by using the group table.

### 4 The extended r-confidential indexing

With Zerber indexing scheme, the adversary on index server only sees the combined posting list length of the merged terms and cannot determine an individual term’s document frequencies (DF). However search and ranking functionalities could be restricted by the full lack of the document frequency. In our extension the dictionary stores not only the terms and pointers to the postings list for the terms, but some data related to document frequencies.

The original DF values cannot be stored because of sensitive information as mentioned above. The first idea to anonymize DF values is the value quantification by uniform step-size [1]. At this quantification the frequent terms with large DF values could be easily distinguished. Our solution for anonymization of DF is based on Zipf’s discovery. Zipf’s law [14] states that given some text corpus of natural language, the frequency of any word is inversely proportional to its rank in the frequency table. Thus the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc. Zipf’s Law is demonstrated with the "The Adventures of Sherlock Holmes by Sir Arthur Conan Doyle", where the occurrences of all terms give Zipfian distribution, that can be seen in Fig. 3, where on the horizontal axe the terms are in decreasing order based on their frequency (the top 5 words of are: 'the', 'I', 'and', 'to', 'of').

![Fig. 3 Zipfian distribution](image)

Let the anonymized value of DF is \( ADF=[\log_a(DF)] \), where \([x]\) denotes the rounded down value of \( x \), so the largest integer that does not exceed \( x \). The parameter \( a \) of logarithmic function is the anonymization parameter: larger value gives larger anonymity, e.g. 10 is a good value to make the original values almost indistinguishable. ADF values are stored in the dictionary. The inverse document frequency (idf) of term \( t_i \) according to [2] is:

\[
idf_i = \log \frac{N}{DF_i} = \log N - \log DF_i
\]

where \( N \) is the number of documents in the corpus. We introduce anonymized inverse document frequency (aidf) of term \( t_i \) using anonymization parameter \( a \) as the base of logarithmic function:

\[
aidf_i = [\log_a N] - [\log_a DF_i] = AN - ADF_i
\]

where \( AN=[\log_a(N)] \). Then aidf \( i \) is given as a non-negative integer value.

In the information retrieval literature [2], the weight of term \( t_i \) in document \( d_j \) can be calculated by the well-known tf-idf formula as:
where $tf_{i,j}$ is the normalized term frequency, i.e. the number of the term $t_i$ occurrences in document $d_j$ divided by the number of the all terms in the document (length of document $d_j$). In our case the weight of term $t_i$ in document $d_j$ is calculated as:

$$w_{i,j} = tf_{i,j} \cdot aidf_i = tf_{i,j} \cdot \left(AN - ADF_i\right)$$

Thereby the ranking can be based on decreasing order of the similarity score between the document $d_j$ and multi-term query $q$. Calculating the weight $w_{i,q}$ for $i$th term of $q$ as

$$w_{i,q} = tf_{i,q} \cdot aidf_i = tf_{i,q} \cdot \left(AN - ADF_i\right)$$

we can compute the cosines similarity [2] between the document $d_i$ and multi-term query $q$ containing $n$ terms as

$$score(d_j,q) = \frac{d_j \cdot q}{\|d_j\| \|q\|} = \frac{\sum_{i=1}^{n} w_{i,j} \cdot w_{i,q}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \cdot \sqrt{\sum_{i=1}^{n} w_{i,q}^2}}$$

Another well-known formula, Okapi BM25 formula can be also used for calculation of the similarity score between the document $d_i$ and multi-term query $q$ containing $n$ terms as

$$score(d_j,q) = \sum_{i,q} aidf_i \cdot \frac{(k_1 + 1) \cdot \cdot \cdot tf_{i,j}}{k_1(1 - b + b \cdot \frac{L_d}{L_{ave}}) + tf_{i,j}}$$

where the query-dependent $aidf_i$ is:

$$aidf_i = \log \frac{a^{AN} - a^{ADF_i} + 0.5}{a^{ADF_i} + 0.5}$$

which is derived from Okapi’s $idf$ formula:

$$idf_i = \log \frac{N - DF_i + 0.5}{DF_i + 0.5}$$

In the equation (11) $L_d$ is the length of document $d$, $L_{ave}$ is the average length of documents in the corpus, $k_1$ is a positive tuning parameter between 1.2 and 2, furthermore $b$ is equal to 0.75.

The extended solution is a generalization of the Zerber indexing scheme. Introduction of the anonymization parameter $a$ gives large space for tuning the scheme. If the parameter $a$ is large enough, then $aidf$ values become constant, as in Zerber scheme. Using an appropriate anonymization value $a$, some $aidf$ values are differentiated, that will improve the quality of ranking and will somewhat increase the probability of the successful Attack 2.

By the extension of the Zerber indexing scheme the proposed method achieves two goals: (i) the ability to use more terms in query and (ii) a better ranking. The normal users, who do not want to attack the index, will be satisfied because of better ranking due to the usage of the query-dependent $aidf$, while only a negligible larger possibility of successful attack is derived for the adversary.

5 Conclusion

An adversary can attack on the set of documents, even if the documents are secured and the access is controlled. An index structure is needed to support efficient search and retrieval, but it can lead to information leakage, an adversary can draw probabilistic inference about the contents of a document collection. Zerber [12] and Zerber’s [13] systems provide a confidential index structure and solve ranking of retrieved documents for the query, but only for single-term queries and constant inverse document frequency ($idf$) factor is used, which leads to a relatively poor accuracy in ranking. The proposed extended solution provides a better ranking using an anonymization parameter and query-dependent anonymized inverse document frequency ($aidf$) factors, furthermore gives possibility of multi-term queries.

References


