ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

A Complex Query Based Private Searching on Streaming Data

Megha Babu[1], Sebastian George[2]

M.Tech Student, Dept. of CSE, VJCET, Vazhakulam, Kerala, India[1]

Asst. Professor, Dept. of CSE, VJCET, Vazhakulam, Kerala, India[2]

ABSTRACT: The users need to keep the search criteria classified because the program residing in the public server may falls into enemy's hands. A complex query based private searching on streaming data help to achieve this goal. The users could efficiently search for documents that satisfy secret criteria (such as presence or absence of a hidden combination of keywords) under various cryptographic assumptions. It is a process to dispatch to a public server a program, which searches streaming sources of data without revealing search criteria. It searches for documents from streaming data on the basis of keyword frequency, such that the frequency of a keyword is required to be higher or lower than a given threshold. This form of query helps in finding more relevant documents. Here it searches for documents from streaming data privately with minimum communication complexity. It has many applications for the purpose of intelligence gathering. For example, one can use this technique to find documents that contain some optional words, relevant words and absence of some words, in a single search, with minimum communication complexity, without revealing the keywords.

KEYWORDS: fully homomorphic encryption, Private searching on streaming data

I. INTRODUCTION

The problem of private searching on streaming data is motivated by intelligence community. They collect useful information from streaming data flowing through a server. The data is potential useful and raises a red flag is classified and satisfies secret criteria. The challenge is how to keep the search criteria classified because the program residing in the public server may falls into enemy's hands. It has applications for the purposes of intelligence gathering. For example, one can use the technique to find if any of hundreds of passenger lists has a name from a list of terrorists and, if so, to find his itinerary without revealing the secret terrorists list.

One of the tasks of the intelligence community is to collect information from streaming data flowing through a server. The streaming data includes one document/message/packet at a time. The streaming data sources examples can be packet traffic on network routers, on-line news feeds, internet chat-rooms, or potentially terrorist related blogs or websites. Most of the streaming data is immediately dismissed and dropped from the server. Only for a limited period, it will reside in the server. A naive method for private search is to download the entire resource to the client machine and perform the search locally. This solution is infeasible due to the large size of the data set to be searched, the limited bandwidth between the client and the remote host, or to the unwillingness of the other party to disclose the entire resource to the client. The criteria is a set of keywords that raises a red flag. Clearly, a preferable solution is to sieve all these data streams at their sources (on the same computers or routers where the stream is generated or arrives in the first place). The issue is how can this be accomplished while keeping the search criteria classified, because the computer where the sieving program executes may falls into enemy's hands.

The existing solutions for private searching [1], [3], [5], [9], [12] on streaming data have not considered combined keyword search i.e., the private search that supports the conjunction, disjunction and pure negation at the same time. Here the new private query searches for documents from streaming data based on combined keyword search. For example, find documents that satisfy secret criteria (such as presence or absence of a combination of hidden keywords). And the existing solutions [1], [3], [5], [9], [12] have high communication complexity. In order to retrieve the documents with minimum communication complexity, the adaptive arithmetic compression is used. It is based on fully homomorphic encryption schemes.

II. RELATED WORK

The section carried out detailed studies on private search [6], [8], [10], streaming data search [4], [7], [11] and private search on streaming data [1], [3], [5], [9], [12]. Usually private search is done in small messages, file names etc. Less no of works concentrate on search in entire document. Reference [4] gives the algorithms for decision tree induction from data streams. It focus on numerical attribute, developed a data structure to insert and calculate best split point efficiently. The One-pass algorithm reduces the processing time and keeps the same tree size and accuracy. The demerit of this algorithm is that it cannot be very accurate. The work [7] gave a new method for privately detecting classified file signatures on untrusted systems. Paillier encryption is used to construct a simple bitmask identifying all classified signatures present on a particular host. No information can be leaked because all operations are over encrypted ciphertexts, even in compromised or untrusted contexts. It doesn't check the entire file, only the file signature is checked. Reference [11] addresses a conjunction operator for private stream searching. It is a system of cryptographic methods and thus preserves the confidentiality of the search criteria and the result. Conjunction operator widens the search capability, which is a bitwise summation of hashed keyword values to reference an encrypted entry in the filter. But this method is suitable only for retrieval of fields from record, not is not suitable for retrieval of documents. The work [6] is based on Quantum Private Query (QPQ) protocol. It ensures data privacy. The QPQ protocol introduced a cheat sensitive strategy which addresses user data privacy. The demerit is that to obtain high level of privacy, large amount of fake query is required. Thus will increase the communication cost. Reference [8] offers a model to achieve balance among the conflicting requirements of security, and practical efficiency, but it doesn't fulfil anonymity and authorization requirements. Reference [10] gives Obfuscation-based private web search (OB-PWS) solutions, which allow users to search for information in the Internet while concealing their interests. The basic privacy mechanism is the automatic generation of dummy queries that are sent to the search engine along with users' real requests. The main drawback of this work is that the automatic generation of dummy queries that are sent to search engine will increase cost. The two solutions for private searching on streaming data are given in [3]. One is based on the Paillier cryptosystem and allows to search for documents satisfying a disjunctive condition. Another is based on the Bonehet cryptosystem and can search for documents satisfying an AND of two sets of keywords. The work [5] also gave a solution to search for documents satisfying a condition k_1 v k_2 ...., $k_{|K|}$. Like the idea of [3], an encrypted dictionary is used. However, rather than using one large buffer, store the matching documents in three buffers and retrieved them by solving linear systems. So documents are retrieved without collisions. The solution in [9] search for documents containing more than t out of n keywords. Buffer keeps at most m matching documents. Searching for documents containing one or more classified keywords like [3], [5] could be achieved. In [12], the author presents single-server computationally private PIR scheme. This method can be adapted to the private domain. The main drawback is that it increases communication and computation costs. The work [1] proposed constructions for conjunctive, disjunctive, and complement query, which searches for documents from streaming data based on keyword frequency, such that a number of times that the search term appears is lower or higher than a given threshold It could be found that, on the whole, the communication complexity is higher for the given works. The complex query based keyword search is not considered in the private searching on streaming data.

III. SYSTEM OVERVIEW

A complex query based private search on streaming data has mathematical properties suitable for implementations of private stream search. This version of private search relies on additively fully homomorphic encryption scheme. Like the streaming model given in [1] and [3], [5] consider a universe of words W = {0, 1}, and a dictionary $D \subset W$ with $| D | < \infty$. Document M just to be an ordered, finite sequence of words in W and S, a stream of documents is to be any sequence of documents. A set of keywords is to be any subset $K \subset W$. A query Q over a set of keywords K, denoted as Q_k, is a logical expression of keywords in K.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

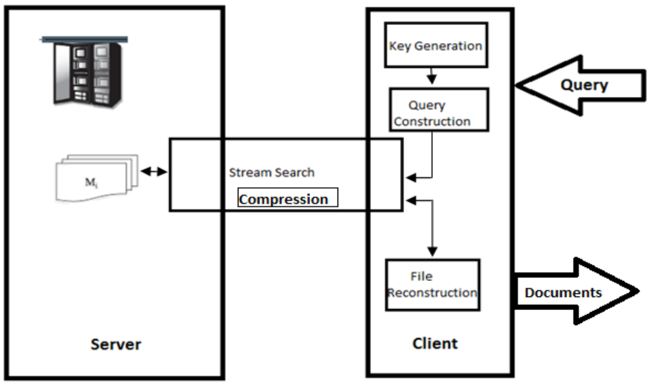

Fig 1. Model for complex query based private searching on streaming data.

The user gives the keywords for private search and retrieves the relevant documents. The construction is composed of the following steps as discussed in the following sections. The last three steps are processed iteratively, until the results are obtained. i.e., the query constructions are considered. The disjunctive stream search occurs. The result, i.e., encrypted buffer is compressed and given to client. The decompression and decryption of buffer occurs in client. Next conjunctive stream search happens in the result of disjunctive stream search. Likewise negation stream search occurs in conjunctive stream search result. Thus the final buffer is obtained. Then according to the buffer value, the corresponding documents are retrieved from server.

A. Key Generation.

In Key Generation, chooses a security parameter as low, medium or high. According to the parameter chosen, the security of the system varies. Then choose the parameters δ, ρ, η, τ satisfying security parameters. Chooses a random odd η bit integer p from $(2Z+1)\cap(2^{\eta}-1, 2^{\eta})$ as the private key s_k. Randomly chooses $q_0, q_1, \ldots, q_\tau$ from $[1, 2^{\delta}/p)$ subject to the condition that the largest q_i is odd and relabel $q_0, q_1, \ldots, q_\tau$ so that q_0 is the largest. Randomly chooses $r_0, r_1, \ldots, r_\tau$ from $Z \cap (2^{-\rho}, 2^{\rho})$ and sets $x_0 = q_0 p + 2r_0$ and $x_i = q_i p + 2r_i \bmod x_0$. The public key is $pk = < x_0, x_1, \ldots, x_\tau >$.
Encrypt(p_k, m). To encrypt $m \in \{0, 1\}$, chooses a random subset $S \subset \{1, 2, \ldots, \tau\}$ and a random integer r from $(2^{-\rho}, 2^{\rho})$ and outputs
$c = \varepsilon(m) = m + 2r + \sum_{i \in S} x_i (\bmod\ x_o)$ eq. (1)

B. Query Construction.

The Query Constructions are generated for the given query, which includes conjunction, disjunction and negation queries.

1) Query Construction Q1:

Construct the Query Construction Q1 for the first set of keywords, i.e., disjunctive query Q_{k1}. Assume that the public dictionary $D = \{m_1, m_2, \ldots, m_{|d|}\}$, keywords $K = \{k_1, k_2, \ldots, k_l\} \subset D$, $d = \log_2|M|$ where $|M|$ is the maximum number of words the document M may contain, then Q1 consists of the dictionary D, the query sign (denoted as 00) and array of ciphertexts

$$\hat{D} = \{\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_{|D|}\}, \text{ where } \hat{m}_i = \varepsilon_{pk}(u_i) \text{ and}$$

$$u_i = \begin{cases} \text{frequency threshold for } k_j, & \text{if } m_i = k_j \in K \\ 2^d - 1, & m_i \notin K \end{cases} \quad \text{eq. (2)}$$

Because the document M contains at most 2^d -1 words, the frequency of any word in M will be less than 2^d - 1. So that the word frequency in M is never more than the threshold. Assume $u_i = (a_{i1}, a_{i2}, \ldots, a_{id})_b$, where $a_{ij} \in \{0, 1\}$, then $\hat{m}_i$ =

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

$\varepsilon_{pk}(u_i) = (\varepsilon_{pk}(a_{i1}), \varepsilon_{pk}(a_{i2}), \ldots, \varepsilon pk(a_{id}))$. The array of ciphertexts is $\hat{D}$, has l encryptions of frequency threshold of matching document i.e., l and $|D| - l$ encryptions of $2^d - 1$.

2) Query Construction Q2:

Construct the Query Construction Q2 for conjunctive query Q_{k2}. Like the previous query construction, assume that the public dictionary and keywords, $d = \log_2|M|$. Then Q1 consists of the dictionary D, the query sign (denoted as 00) and array of ciphertexts
$\hat{D} = \{ \hat{m}_1, \hat{m}_2, \ldots, \hat{m}_{|D|} \}$
Where $\hat{m}_i = \varepsilon_{pk}(u_i)$ and

$$u_i = \begin{cases} \text{frequency threshold for } k_j, & \text{if } m_i = k_j \in K \\ 0, & m_i \notin K \end{cases} \qquad \text{eq. (3)}$$

Assume $u_i = (a_{i1}, a_{i2}, \ldots, a_{id})_b$, where $a_{ij} \in \{0, 1\}$, then $\hat{m}_i = \varepsilon_{pk}(u_i) = (\varepsilon_{pk}(a_{i1}), \varepsilon_{pk}(a_{i2}), \ldots, \varepsilon_{pk}(a_{id}))$. The array of ciphertexts is $\hat{D}$, has m encryptions of frequency threshold of matching document i.e., 1 and $|D| - m$ encryptions of 0.

3) Query Construction Q3:

Construct Query Construction Q3 for negation query Q_{k3}. A negation query over keywords $K = \{k_1, k_2, \ldots, k_n\}$. Like the previous Query Constructions, the public dictionary $D = \{m_1, m_2, \ldots, m_{|d|}\}$. Let $k_1, k_2, \ldots, k_n = K \subset D$. And, $d = \log_2|M|$. Q3 consists of the dictionary D, query sign (denoted as 11) and an array of ciphertexts
$\hat{D} = \{\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_{|D|}\}$, where $\hat{m}_i = \varepsilon_{pk}(s_i)$ and

$$s_i = \begin{cases} 1, & \text{if } m_i \in \{k_1, k_2, \ldots, k_n\} \\ 0, & m_i \notin K \end{cases} \qquad \text{eq. (4)}$$

The array of ciphertexts has n encryptions of frequency thresholds i.e., n and $|D|$- n encryptions of 0.

C. *Stream Search.*

1) Disjunctive Stream Search:

Consider disjunctive query construction. It constructs a buffer B with ml boxes, each of them is initialized with $\varepsilon_{pk}(0)$, where m is the maximum number of matching documents kept in the buffer and l is the size of the document. Then constructs the encryption of the two's complement of $-u_i$ (denoted as $-\bar{u}_i$) with $\hat{m}_i = \varepsilon_{pk}(u_i)$, that is, $\varepsilon_{pk}(\bar{u}_i) = (\varepsilon_{pk}(1), \varepsilon_{pk}(a_{i1}) + 1, \ldots, \varepsilon_{pk}(a_{id} + 1) \oplus \varepsilon_{pk}(1)$
The leftmost bit of the two's complement of an negative integer is 1 or 0. On receiving an input document $M = (m_1, m_{2,}, \ldots, m_l)_b$ from the stream S, need to determine that the document is a matching or not, So homomorphically compute a ciphertext $\varepsilon_{pk}(f_0)$ such that a document is matching, if $f_0 = 1$ and 0 otherwise. And continue with the following steps:

a) *Word Collection 1 :* First collects

$$\widehat{H} = \{m_i, f(m_i) \mid m_i \in M \cap D\}$$

where $f(m_i)$ is the frequency of m_i in each document M. $\hat{H}$ is the set of words common to document M and the dictionary D and the corresponding count of matching words in document. Next, and the encryption of the two's complement of $\overline{f(m_i)} = (b_{i1}, b_{i2}, \ldots, b_{id})_b$, denoted as $f(m_i)$, for each $m_i \in \hat{H}$, that is,
$\varepsilon_{pk}(f(\overline{m_i})) = \varepsilon_{pk}(0), \varepsilon_{pk}(b_{i1}), \varepsilon_{pk}(b_{i2}), \ldots, \varepsilon_{pk}(b_{id}))$

b) *Frequency Comparison 1 :* For each $m_i \in \hat{H}$, homomorphically compares the frequency $f(m_i)$ and u_i by computing

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

$$\varepsilon_{pk}\,(\overline{f(m_i)}+ -\bar{t}_i)$$
$$= \varepsilon_{pk}\,(\overline{f(m_i)}) \oplus \varepsilon_{pk}\,((-\bar{t}_i))$$
$$= (\varepsilon_{pk}\,(f_{i0}), \varepsilon_{pk}(f_{i1}), \varepsilon_{pk}(f_{i2}), ..., \varepsilon_{pk}(f_{id})),$$

from which only ε_{pk} (f_{i0}) is extracted. In two's complement system, if $(f_{i0}) = 0$, then $f(m_i) \geq u_i$ and otherwise $f(m_i) < u_i$. Next, computes

$$\varepsilon_{pk}\,(f_0) = \varepsilon_{pk}\,(\vee_{mi\in\hat{H}}\,(f_{io} \oplus 1)) \qquad \text{eq. (5)}$$

by repeatedly using $\varepsilon_{pk}\,(f_{i0} \vee s) = \varepsilon_{pk}\,(f_{i0})+ \varepsilon_{pk}\,(s)+ \varepsilon_{pk}\,(f_{i0})\varepsilon_{pk}\,(s)$. If $f_0 = 1$, then it is considered that there exists i such that $f_{i0} \oplus 1 = 1$ (i.e., $f_{i0} = 1$ and $f(m_i) \geq t_i$). If $m_i \notin K$, then $t_i = 2^d - 1$ and it is not possible that $f(m_i) \geq 2^d - 1$. So it denotes that $m_i \in K$ and $f(m_i) \geq u_i$, the document is a matching document. If $f_0 = 0$, then $f_{i0} \oplus 1 = 0$ (i.e., $f_{i0} = 1$ and $f(m_i) < u_i$) for all $m_i \in M \cap D$. This means that the particular document is not a matching document.

c) *Matching Document 1 :* The state of buffer B is $(b_1, b_2, ...,b_L)$, where $L = ml$. The b_i is an encryption of either 0 or 1, Because the $\varepsilon_{pk}(f_{i0})$ is stored in the buffer. To store the encryption of the document M into the buffer B, position wise adds the result into the buffer B, denoted as

$$B = B + \hat{M}$$

If $f_0 = 1$, then $\hat{M}$, the encryption of the binary linear code of the matching document M, is kept in the buffer B. If $f_0 = 0$, then $\hat{M}$ is the encryption of 0, which has no effect on the buffer B. The encrypted buffer is compressed (as in subsection D) and send to client. Then filter out the documents with $f_{i0} = 0$ (as in subsection E). Next the conjunctive stream search takes place in the documents, which have $f_{i0} = 1$.

2) *Conjunctive Stream Search:*

Consider the Query Construction for conjunctive query. Constructs a buffer B with *ml* boxes, each of them is initialized with $\varepsilon_{pk}(0)$, where m and l are the maximum number of matching documents kept in the buffer and size of the document respectively. It constructs the encryption of the two's complement of $-u_i$ (denoted as $-\bar{u}_i$) with $\hat{m}_i = \varepsilon_{pk}(u_i)$, that is,

$$\varepsilon_{pk}(\bar{u}_i) = (\varepsilon_{pk}(1), \varepsilon_{pk}(a_{i1}) + 1, ..., \varepsilon_{pk}(a_{id} + 1) \oplus \varepsilon_{pk}(1)$$

On receiving an input document M from the stream S, need to determine if M is a matching document or not, so homomorphically compute a ciphertext $\varepsilon_{pk}(f_0)$ such that M is a matching document if $f_0 = 1$ and 0 otherwise. Compute the encryption of $u = \sum_{mi\,\in\,K} u_i$. It proceeds with the following steps

a) *Word Collection 2 :* First collects

$$\widehat{H} = \{m_i, f(m_i) \mid m_i \in M \cap D\}$$

where $f(m_i)$ is the frequency of m_i in the document M, i.e., count of each keyword appears in the document. $\hat{H}$ is the set of common words in the document and the Dictionary. Next constructs the encryption of the two's complement of $\overline{f}(m_i) = (b_{i1}, b_{i2}, ..., b_{id})_b$, denoted as $f(m_i)$, for each $m_i \in \hat{H}$, , that is,
$\varepsilon_{pk}\overline{(f(m_i))} = \varepsilon_{pk}\,(0), \varepsilon_{pk}\,(b_{i1}), \varepsilon_{pk}\,(b_{i2}), ..., \varepsilon_{pk}\,(b_{id}))$
Compute $u' = \sum_{mi\,\in\,\hat{H}} u_i$ and the encryption of the two's complement of $u' = \sum_{mi\,\in\,\hat{H}} u_i = (\alpha_1, \alpha_2, ..., \alpha_d)$, denoted as $\bar{u}'$, that is
$\varepsilon_{pk}\,(\bar{u}') = (\varepsilon_{pk}(0), \varepsilon_{pk}(\alpha_1), \varepsilon_{pk}(\alpha_2),..., \varepsilon_{pk}(\alpha_d))$.

b) *Frequency Comparison 2 :* For each $m_i \in \hat{H}$, homomorphically compares the frequency $f(m_i)$ and the threshold u_i by computing

$$\varepsilon_{pk}\,(\overline{f(m_i)}+ -\bar{u}_i)$$
$$= \varepsilon_{pk}\,(f(m_i)) \oplus \varepsilon_{pk}\,((-\bar{u}_i))$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

$$= (\varepsilon_{pk}(f_{i0}), \varepsilon_{pk}(f_{i1}), \varepsilon_{pk}(f_{i2}), ..., \varepsilon_{pk}(f_{id})),$$

from the computed value only $\varepsilon_{pk}(f_{i0})$ is considered. If $f_{i0} = 0$, then $f(m_i) \geq u_i$ and otherwise $f(m_i) < u_i$.
Next it homomorphically checks if the document M contains all keywords in K by computing

$$\varepsilon_{pk}(u' + \varepsilon_{pk}(-\bar{u}))$$

$$= \varepsilon_{pk}(\bar{u}') \oplus (\varepsilon_{pk}(-\bar{u}))$$

$$= (\varepsilon_{pk}(\delta_0), \varepsilon_{pk}(\delta_1), \varepsilon_{pk}(\delta_2), ..., \varepsilon_{pk}(\delta_d)),$$

from the computed value only $\varepsilon_{pk}(\delta_0)$ is extracted. If $\delta_0 = 0$, then $(u' \geq u)$ and thus $u' = u$ and the searching document contains all keywords in K. If $\delta_0 = 1$, then $u' < u$ and the document does not contain all keywords in K. Next, it proceeds with the following steps,

$$\varepsilon_{pk}(f_0) = \varepsilon_{pk}(\delta_0 \oplus 1) \wedge_{mi \in \hat{H}} (f_{io} \oplus 1)) \qquad \text{eq. (6)}$$

If $f_0 = 1$, then $\delta_0 = 0$ and $f_{i0} = 0$ for all $m_i \in \hat{H}$. If $\delta_0 = 0$ means $\hat{H} \cap K = K$ while $f_{i0} = 0$ for all $m_i \in \hat{H}$ means $f(m_i) \geq u_i$ for all $m_i \in \hat{H}$. It is obvious that $f(m_i) \geq 0$
for all $m_i \notin K$, M is a matching document. If $f_0 = 0$ and $\delta_0 = 1$, the document does not contain all keywords in K. It is not a matching document. If $f_0 = 0$ and $\delta_0 = 0$, M does contain all keywords in K, but there exists i such that $f(m_i) < u_i$. It is not a matching document.

c) *Matching Document 2 :* Buffer B is $(b_1, b_2, ..., b_L)$, where $L = ml$. The b_i is an encryption of either 0 or the $\varepsilon_{pk}(f_{i0})$ is stored in the buffer. The rest of steps is same as the disjunctive threshold query. The encrypted buffer is compressed and send to client (as in section D). This filter out the documents with $f_{i0} = 0$ (as in section E). Next the pure negation stream search takes place in the documents, which have $f_{i0} = 1$.

3) *Negation Stream Search:*
Consider Query Construction Q3. A buffer B is constructed with ml boxes, each of them is initialized with $\varepsilon_{pk}(0)$.

a) *Word Collection 3 :* Collects

$$\widehat{H} = \{m_i, f(m_i) \mid m_i \in M \cap D\}$$

Where $f(m_i)$ is the frequency of m_i in the document M. It constructs the encryption of the two's complement as in disjunctive stream search.

b) *Frequency Comparison 3 :* For each $m_i \in \hat{H}$, the program homomorphically compares the frequency $f(m_i)$and the frequency threshold si by computing
$\varepsilon_{pk}(f(m_i) + (-\bar{s}_i))$
Next check whether the document is a matching document. For that compute

$$\varepsilon_{pk}(f_0) = \prod_{mi \in \hat{H}} (\varepsilon_{pk}(f_{i0}) + \varepsilon_{pk}(1) + \hat{m}'_i \qquad \text{eq. (7)}$$

According to homomorphic properties, if $m_i \in \{k_1, k_2, ..., k_n\}$, then $s_i = 1$ and thus $f_{i0} = 1$, which implies that $f(m_i) < t_i$. And M is a matching document when $f_0 = 1$.

c) *Matching Document 3 :* The buffer B is an encryption of either 0 or 1. This is the final result of stream search, which contains the metadata of relevant documents. This encrypted buffer is compressed and sends to client (as in section D). Then find the documents with $f_{i0} = 1$ (as in section E), which are the final results of search.

D. Compression

Adaptive arithmetic coding encodes the whole message into a single number. It is a fraction n, between $(0 \leq n < 1)$. With the algorithm progresses, it computes the statistics of the symbols. The code associated with the symbols is changed with the re-computation. The encoder has three pieces of data to be checked: The next symbol to be encoded, the current interval, the probabilities the model assigns to each of the symbols that are possible at the stage.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

The probabilities of the message are considered. It narrows the interval following one after the other in a series and it represents the ensemble. When compared to a low probability message, the high probability message narrows the interval. Let N be the size of the alphabet of the data source. Let symbol x's probability be O[x]. Assume the probabilities of each symbol of the data source. Then to each symbol allocate an interval. The width of the interval should be proportional to its probability assigned, and intervals should not overlap with others. This is achieved by using the cumulative probabilities as the two ends of each interval. R[x-1] and R[x] are the two ends of each symbol x. Symbol x is in the range [R[x-1], R[x]). The interval begins with [0, 1) and the interval is subdivided iteratively. For each symbol, the current interval is divided based on the probabilities of the alphabet. The interval to be further proceeded is picked up. Until all symbols in the message have been processed, procedure is continued. For each possible message there is a unique interval assigned because each symbol's interval does not overlap with others. The message is represented within the interval [M, I). It is a single value in the interval as the encoded code. Under normal conditions the left end M is selected. I and M are two ends of the current code interval and S is the interval range. x is the next symbol to be encoded. I is initialized to 0 and M is initialised to1.

```
Adaptive Arithmetic Encoder
M = 0.0; I = 1.0;
While ( (x = getc(input))! = EOF )
{
S = (I - M);
I = M + S _ R[x];
M = M + S _ R[x - 1];
}
Output (L);
```

```
Adaptive Arithmetic Decoder
M = 0.0; I = 1.0;
While ( (x = getc(input))! = EOF )
{
S = (I - M);
I = M + S _ R[x];
M = M + S _ R[x - 1];
}
Output (L);
```

E. File Reconstruction

The buffer received is decompressed in the client side. Using the secret key s_k, the algorithm decrypts the buffer B, sent back by the Server, one box at a time. It retrieves the documents corresponding to the document metadata. To decrypts c, outputs

$$(c \bmod p) \bmod 2 \qquad \text{eq. (8)}$$

In the final file reconstruction, find the documents with $f_{i0} = 1$, which are the results of document search.

IV. PERFORMANCE EVALUATION

The whole experiment is conducted using Reuters 21578 dataset. It is a set of documents from Reuters' 1986 newswire. The performance evaluation is based on the execution time and communication time of the base system and the proposed system. The base system consists of three different basic constructions, which supports simple query based private search. The proposed system supports complex query based with same number of streaming documents and query and compared the systems in milliseconds. It starts from the less number of documents to the more number of documents.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2015

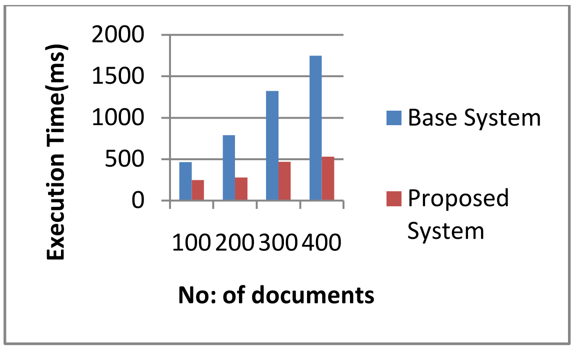

Fig. 2. Graph comparing the execution time of base system and proposed

The execution time of base system is taken by summing up the execution time of simple searches i.e., conjunction, disjunction and negation queries. The execution time of the proposed system is taken by a single complex query search. Fig 2 shows that the proposed system has better execution time than the existing system.

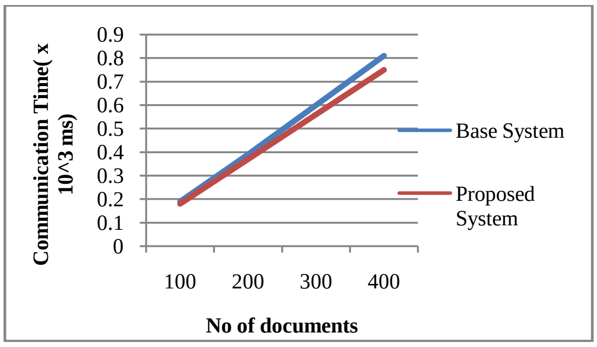

Fig. 3. Graph comparing the communication time of base system and proposed system

For performance evaluation, the method is to compare the communication time of base system and the proposed system. In all the cases the result is that the communication time for proposed system is less compared to the base system. The performance evaluation shows that the compressed results are received faster than uncompressed results. Fig 3 shows that the communication complexity is minimum for proposed system. It is achieved by using adaptive arithmetic compression.

V. CONCLUSION AND FUTURE WORK

A Complex Query Based Private Searching on Streaming Data provides a combined keyword search on streaming documents, whereas the existing system provides only a simple query based private search. It is based on fully homomorphic encryption techniques. The search takes place on the basis of keyword frequency. The complex threshold query is able to search for documents using the set of keywords (presence or absence or combination of hidden keywords) by letting the threshold t = 1. More accurate documents are retrieved in a single search. Execution time of the proposed system is better than the existing system. Adaptive arithmetic compression reduces the communication complexity and retrieves the documents with minimum communication complexity. Future works could include methods which could further reduce the cost in resources like communication and computation.

REFERENCES

1. Xun Yi, Elisa Bertino, Jaideep Vaidya, and Chaoping Xing, "Private Searching on Streaming Data Based on Keyword Frequency", IEEE Trans. dependable and secure computing, vol. 11, no. 2, March/April 2014.
2. Jiaji Wu, Minli Wang, Jechang Jeong, Licheng Jiao, "Adaptive-distributed arithmetic coding for lossless compression", IEEE International Conference on Network Infrastructure and Digital Content, Sept. 2010.
3. Rafail Ostrovsky and W. Skeith,"Private Searching On Streaming Data", j.Cryptology, vol 20, no. 4, pp. 397-430, 2007.
4. Tao Wang, Zhoujun Li, Yuejin Yan, Huowang Chen and Jinshan Yu, "An Efficient Classification System Based on Binary Search Trees for Data Streams Mining", IEEE Int'l Conf. Systems, 2007.
5. J. Bethencourt, D. Song, and B. Water, "New Techniques for Private Stream Searching," ACM Trans. Information and System Security, vol. 12, no. 3, pp. 1-32, 2009.
6. Vittorio Giovannetti and Lorenzo Maccone, "Quantum Private Queries: Security Analysis", IEEE Trans. Information theory, vol. 56, no. 7, July 2010.
7. John Solis, "Private searching for sensitive _le signature", IEEE Conf. Security and Cryptography (SECRYPT), 2011.
8. Raykova. M., Ang Cui, Binh Vo, Bin Liu, Malkin. T., Bellovin, S.M., Stolfo S.J., "Usable, Secure, Private Search", IEEE. Security Privacy, 2012.
9. X. Yi and C.P. Xing, "Private (t, n) Threshold Searching on Streaming Data," Proc. Int'l Conf. Social Computing Privacy, Security, Risk and Trust (PASSAT '12), pp. 676-683, 2012.
10. Balsa. E., Troncoso. C., Diaz. C., "OB-PWS: Obfuscation-Based Private Web Search", IEEE. Security and Privacy (SP), 2012.
11. Oehler. M., Phatak, D.S., "A Conjunction for Private Stream Searching", IEEE Int'l Conf. Social Computing (SocialCom), 2013.
12. Fanti. G., Finiasz. M., Ramchandran. K. , "One-Way Private Media Search on Public Databases", IEEE. Signal Processing, 2013

BIOGRAPHY

Megha Babu is a M.TECH Student in CSE, Viswajyothi College of Engineering & Technology, Kerala, M.G University, India.

Mr. Sebastian George is working as Assistant Professor at Department of CSE, Viswajyothi College of Engineering & Technology, Kerala, M.G University, India.