ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

EKST: Efficient Keyword Searching Technique for Encrypted Data in CipherCloud

V. Sravan Kumar Reddy
M. Tech (SE), CSE Department,
ASCET, Andhrapradesh, India,
vsravankumarreddy@gmail.com

Professor C. Rajendra
Head, CSE Department,
ASCET, Andhrapradesh, India,

***Abstract*— cloud, coming days is accustomed, more and more receptive data are being centralized into the cloud. In cloud, we make perceptive information habitually have to be ciphertext before outsourcing because of protecting of data privacy; it makes efficient information exploitation a very tough assignment. While traditional searchable ciphertext schemes permit a consumer to securely seek over ciphertext information through keywords and selectively reclaim records of attention, these techniques bear only precise keyword search. That is, here is lenience of trivial typos and configure inconsistencies which, on another hand, are distinctive consumer penetrating activities and happen very recurrently. This momentous negative aspect makes existing techniques incongruous in cloud computing as it significantly affects system usability, interpretation user penetrating experiences very testing and system efficacy very low.**

In this paper, we proposing EKST an efficient keyword searching technique as a solution and solve the crisis of searching the fuzzy keyword over encrypted cipharcloud data while maintaining keyword confidentiality. The proposed searching technique deeply enhances system usability by presentencing the matching files when user's searching inputs exactly match the predefined keywords or the closest possible matching files based on keyword comparison semantics, when *exact* match fails. Our EKST greatly reduces the storage and depiction overheads, we develop revise space to enumerate fuzzified keywords similarity and implement a method on constructing fuzzified keyword sets, Through accurate security study, we show that our proposal is secure and privacy-preserving.

***Keywords*: Keyword Searching, fuzzy keywords, cipharcloud**

I. INTRODUCTION

The Cloud Computing is added common to centralize the perceptive data in to the cloud, for example emails data, important documents of government, health data records and etc. The information owners can calmed from encumber of information storage and maintenance by storing their data in to the cloud storage so as to enjoy the on-demand high quality data storage service. The data stored in outsource cloud may risk because actuality the data owners and cloud server are no in the same trusted province, this may causes the data at risk. The cloud follows that responsive information usually should be encrypted prior to outsourcing for information privacy and warfare unwanted accesses. But, encryption made helpful information exploitation a very difficult task given that there could be a mammoth amount of outsource information records. Furthermore, in Cloud Computing, information holders may share their outsourced information with a outsized number of users.

The different users in the cloud storage may interest to recover the detailed information files they want in a specified time. In cloud computing the trendy technique to reclaim the information is keyword searching, retrieving encrypted information files back which is utterly unfeasible in cloud computing but the keyword based vanished this scenario. The users can recoup their information in the cloud by using he keyword searching techniques, these techniques are mostly applied in plain text search scenarios, like Google [1].

Unhappily, the plaintext keyword search techniques are not suitable for the cloud computing, because the encrypted data won't allow performing the search. The keywords contain significant information related to the data, so the encryption needs to protect the keyword privacy so secure searching techniques are needed in cloud computing. In recent years few securely searchable encryption methods [2] on encrypted data have been developed. The securely searchable encryption scenarios regularly follow by indexing the each keyword in encrypted data file and by associating the indexed file with the keyword.

The searchable encryption techniques proposed in previous are not suitable for cloud computing to perfume the searches, because they support only the exact keyword search. That is, there is no lenience of slight typos and plan inconsistencies. It is fairly frequent that users' probing keywords would not exactly match those fixed keywords due to the possible typos, such as Illiterate and Iliterate representation inconsistence, such as PO BOX and P.O. Box. and/or her lack of exact knowledge about the data. The adolescent mode to support fuzzy keyword search is by using simple spell check technique. But, this approach does not completely fulfill the requirement to solve the problem and sometimes can be unsuccessful due to, it requires added interface of user to resolve the right word from the candidates generated by the spell check algorithm, this may gratuitously costs user's extra computation effort; on the other hand, in case that user incorrectly types some other keywords by mistake, the spell check algorithm would not even work at all, as it can never discriminate between two actual valid words. Thus, the drawbacks of existing approach indicate the important need for new techniques that support searching elasticity, tolerating both minor typos and format inconsistencies.

As a solution, we are proposing EKST and efficient keyword search technique yet privacy preserving in Cloud Computing. This keyword search technique really enhances the system usability by chronic the matching data files when

user searching input keywords exactly match the redefined keywords or by using the keyword similarity semantics when the match fails. Importantly we use distance editing to enumerate keywords similarities and impremted novel technique. i.e., a wild card based technique, for the construction of fuzzy keyword sets. The wild card technique avoids the need for specifying the one keyword after one keywords and the size of the keyword sets is significantly reduced. Based on the constructed fuzzy keyword sets, we propose an efficient fuzzy keyword search scheme. Through careful security scrutiny, we show that the proposed solution is secure and privacy-preserving, while correctly realizing the goal of fuzzy keyword search.

II. Related Work

Plaintext fuzzy keyword search, today the significance of fuzzy keyword search has established concentration in the circumstance of plain text searching in data repossession area [11]. They proposed a solution to this problem in the conventional retrieval model by allowing data holder to search without using try-and-get approach for verdict pertinent data based on estimated keyword matching. At the first glimpse, it seems probable for one to straightly apply these keyword matching algorithms to the milieu of searchable encryption by computing the trapdoors on a character base within an alphabet. However, this unimportant creation suffers from the glossary and data attacks and fails to achieve the search privacy.

Searchable encryption, In [2][10] extensively considered about the predictable searchable encryption in the framework of cryptography. Most of them described on effectiveness improvements and security description formalizations. *Song* [3] projected the initial structure of searchable encryption, in his proposal every word in the document is treated as keyword and is encrypted separately less a particular two layered structure. The Bloom filters to build the indexes for the data in the documents are proposed by Gosh in [4]. To achieve the more proficient search system two authors Chang [7] and Curtmola [8] proposed an approach which single encrypted hash table index is built for the entire document. The Boneh [5] proposed a public key based searchable encryption method as a complementary approach of Chang and Curtomola with an analogous scenario to that of Song model. The techniques described in this session or suitable for the plaintext encryption, but not for the Cloud Computing, Because Cloud doest not support the plaintext searchable encryption techniques. All these presented methods are for the plain text by matching accurate keywords.

Private matching in [14] is another related notion, and the author has been considered typically in the circumstance of secure shared computation to let multiple parties compute some task of their own data collaboratively without revealing their data to the others. These functions could be crossroads or estimated personal matching of two sets, etc. The private information retrieval [15] is an regularly used method to regain the similar items secretly, which has been generally functional in information retrieval from database and usually incurs unexpectedly computation intricacy.

III. Problem Formulation

A. EKST Model

Our proposed cloud data system consisting of three major roles cloud data owner, user and server. Given a group of n encrypted data files $DF = (F_1, F_2, \ldots, F_N)$ stored in the cloud server, a predefined set of distinct keywords $KW = \{kw_1\ kw_2, ..., kw_p\}$, the endorsed users will get a search service from the cloud server to search over the encrypted data DF. We assume the endorsement between the data owner and users is appropriately done. An endorsed user types in a request to selectively regain data files of his/her interest. The data files and the search request association will be done by the server, in the association process every file is indexed by a file ID and associated to a set of keywords. The EKST fuzzy keyword search scheme returns the search results according to the following rules:

i. *The server returns the datafiles containing the keywords; if the user's input words perfectly matches the predefined set of keywords.*
ii. *Depends on predefined similarity semantics the server will return the closet possible results; if there exist typos and/or format inconsistency in the searching input.*

Structure of fuzzy keyword search is shown in the Fig. 1.

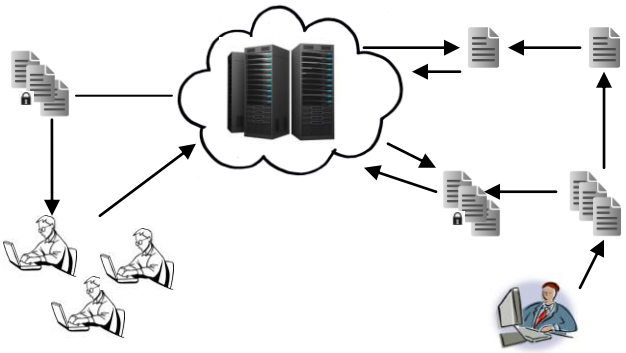

Figure 1. *Structure of fuzzy keyword search*

B. Hazard Model

We reflect on a semi-trusted server. While performing keyword based search over DF the cloud server could attempt to obtain other receptive data form the data user's requests even if data files are encrypted, thus the search should be conducted in a secure manner that allows data files to be strongly retrieved while illuminating as little data as potential to the cloud server. In our model, when designing EKST, we will track the security description presented in the traditional searchable encryption. The security of remotely stored data files and index and pattern of queries is more important.

C. Devise Goals

In EKST, we tackle the problem of underneath competent yet privacy preserving fuzzy keyword search services over encrypted data in the cloud. Specifically, we have the subsequent goals: i) to discover new method for constructing storage competent fuzzy keyword sets; ii) to design proficient and valuable fuzzy search method based on

the structured fuzzy keyword sets; iii) to validate the security of the proposed scheme.

D. Preliminaries

Edit Distance There are so many methods to determine the fuzzy keyword similarity. In this paper, we remedy to the well studied edit distance [16] for our purpose. The edit distance $d(kw_1, kw2)$ between two words kw_1,kw_2 is the number of operations required to transform one of them into the other. The three primitive operations are 1) Substitution: changing one character to another in a word; 2) Deletion: deleting one character from a word; 3) Insertion: inserting a single character into a word. Given a keyword kw, we let $S_{w,d}$ denote the set of words kw' satisfying $ed(kw,kw') \leq d$ for a certain integer d.

Fuzzy Keyword Search Using edit distance, the definition of fuzzy keyword search can be formulated as follows: Given a collection of n encrypted data files $DF = (F_1, F_2, \ldots, F_N)$ stored in the cloud server, a set of distinct keywords $KW = \{kw_1, kw_2, \ldots, kw_p\}$ with predefined edit distance d, and a searching input (kw, k) with edit distance k $(k \leq d)$, the execution of fuzzy keyword search returns a set of file IDs whose corresponding data files possibly contain the word w, denoted as FID_w: if $kw = kw_i \in KW$, then return FID_{wi}; otherwise, if $kw \notin KW$, then return $\{FID_{wi}\}$, where $d(kw,kw_i) \leq k$. Note that the above definition is based on the assumption that $k \leq d$. In fact, d can be different for distinct keywords and the system will return $\{FID_{wi}\}$ satisfying $d(w,w_i) \leq \min\{k, d\}$ if exact match fails.

IV. The Straightforward Approach

Prior to describing our structure of fuzzy keyword sets, we first put forward a straightforward method that achieves all the functions of fuzzy keyword search, which aims at giving an overview of how fuzzy search scheme works over encrypted data.

Assume $\prod = (Stup\ (1^\lambda), Ec\ (sk, \cdot), Dc\ (sk, \cdot))$ is a symmetric encryption method, where sk is a secret key, $Setup\ (1^\lambda)$ is the setup algorithm with security parameter λ, $Enc(sk, \cdot)$ and $Dc(sk, \cdot)$ are the encryption and decryption algorithms, respectively. Let T_{kwi} denote a trapdoor of keyword kw_i. Trapdoors of the keywords can be realized by applying a one-way function f, which is similar as [2], [4], [8]: Given a keyword kw_i and a secret key sk, we can compute the trapdoor of kw_i as $T_{wi} = f(sk,kw_i)$.

The scheme of the fuzzy keyword search goes as follows:

We begin by constructing the fuzzy keyword set $S_{kwi,d}$ for each keyword $kw_i \in W$ $(1 \leq i \leq p)$ with edit distance d. The intuitive way to construct the fuzzy keyword set of kw_i is to enumerate all possible words kw'_i that satisfy the similarity criteria $ed(kw_i,kw'_i) \leq d$, that is, all the input words with edit distance d from w_i are listed. For example, the following is the listing variants after a substitution operation on the first character of keyword CASTLE: {AASTLE, BASTLE, DASTLE, · · ·, YASTLE, ZASTLE}. Based on the resulted fuzzy keyword sets, the fuzzy search over encrypted data is conducted as follows:

i. *To build an index for kw_i, the data owner computes trapdoors $T_{kwi} = f(sk,kw'_i)$ for each $kw'_i \in S_{kwi,d}$ with a secret key sk shared between data owner and authorized users. The data owner also encrypts FID_{kwi} as $Enc(sk, FID_{kwi}kw_i)$. The index table $\{(\{T_{kwi}\}_{kwi \in S_{kwi,d}}, Enc(sk, FID_{wi}kw_i))\}_{wi \in W}$ and encrypted data files are outsourced to the cloud server for stroage;*
ii. *To search with kw, the authorized user computes the trapdoor T_{kw} of kw and sends it to the server;*
iii. *Receiving the search request T_{kw}, the server compares it with the index table and returns all the possible encrypted file identifiers $\{Enc(sk, FID_{wi}kw_i)\}$ according to the fuzzy keyword definition. The user decrypts the returned results and retrieves pertinent files of interest.*

V. Constructions of Effective Keyword Search in Cloud

The main proposal behind our EKST is two-fold:

i. *building up fuzzy keyword sets that incorporate not only the exact keywords but also the ones differing slightly due to minor typos, format inconsistencies, etc.;*
ii. *designing an efficient and secure searching approach for file retrieval based on the resulted fuzzy keyword sets.*

A. Advanced Technique for building Fuzzy Keyword Sets

The structure of fuzzy keyword sets aims to improve the straightforward method, for both the storage and efficiency in the searching we now proposing an advanced technique for the straightforward method improvement. To elaborate the proposed advanced technique we focus on the case of edit distance $d = 1$ with out loss of simplification.

For better values of d, the analysis is similar. Note that the procedure is carefully designed in such a way that while suppressing the fuzzy keyword set, it will not affect the search accuracy.

Wildcard-based Fuzzy Set Construction In the above straightforward approach, all the variants of the keywords have to be listed even if an operation is performed at the same position. Based on the above scrutiny, we proposed to use a wildcard to signify edit operations at the same position. The wildcard-based fuzzy set of w_i with edit distance d is denoted as $S_{kwi,d}=\{S'_{kwi,0}, S'_{kwi,1}, \cdots, S'_{kwi,d}\}$, where $S'_{kwi,\tau}$ denotes the set of words kw'_i with τ wildcards. Note each wildcard represents an edit operation on w_i. For example, for the keyword *CASTLE* with the pre-set edit distance 1, its wildcard-based fuzzy keyword set can be constructed as $S_{CASTLE,1} = \{CASTLE, *CASTLE, *ASTLE, C*ASTLE, C*STLE, \cdots, CASTL*E, CASTL*, CASTLE*\}$. The total number of variants on CASTLE constructed in this way is only $13 + 1$, instead of $13 \times 26 + 1$ as in the above exhaustive enumeration approach when the edit distance is set to be 1. Generally, for a given keyword kw_i with length ℓ, the size of $S_{kwi,1}$ will be only $2\ell + 1 + 1$, as compared to $(2\ell + 1) \times 26 + 1$ obtained in the straightforward approach. The larger the pre-set edit distance, the more storage overhead can be reduced: with the same setting of the example in the straightforward approach, the proposed technique can help reduce the storage of the index from 30GB to approximately 40MB. In case the edit distance is set to be 2 and 3, the size of $S_{kwi,2}$ and $S_{kwi,3}$ will be $C_{1\,\ell+1} + C_{1\,\ell} \cdot C_{1\,\ell} + 2C_{2\,\ell+2}$ and $C_{1\,\ell} + C_{3\,\ell} + 2C_{2\,\ell} + 2C_{2\,\ell} \cdot C_{1\,\ell}$. In

other words, the number is only $O(\ell d)$ for the keyword with length ℓ and edit distance d.

B. The Efficient Keyword Search Technique

Based on the storage-efficient fuzzy keyword sets, we show how to construct an efficient and effective fuzzy keyword search scheme. The scheme of the fuzzy keyword search goes as follows:

i. *To build an index for w_i with edit distance d, the data owner first constructs a fuzzy keyword set $S_{kw_i,d}$ using the wildcard based technique. Then he computes trapdoor set $\{T_{kw'_1}\}$ for each $kw'_i \in S_{kw_i,d}$ with a secret key sk shared between data owner and authorized users. The data owner encrypts FID_{kw_i} as $Enc(sk, FID_{kw_i}kw_i)$. The index table $\{(\{T_{kw'_1}\}_{k_i} \in S_{kwi,d}, Enc(sk, FID_{kw_i}kw_i))\}_{kw_i} \in kw$ and encrypted data files are outsourced to the cloud server for storage;*

ii. *To search with (kw, k), the authorized user computes the trapdoor set $\{T_{kw'}\}_{kw' \in S_{kw,k}}$, where $S_{w,k}$ is also derived from the wildcard-based fuzzy set construction. He then sends $\{T_{kw'}\}_{kw' \in S_{kw,k}}$ to the server;*

VI. Security Analysis

This section analyzes the accuracy and security of the proposed EKST. At first, we show the correctness of the schemes in terms entirety and reliability.

Theorem 1: The wildcard based technique satisfies both entirety and reliability. Specifically, upon receiving the request of *kw*, all of the keywords $\{kw_i\}$ will be returned if and only if $ed(kw,kw_i) \le k$. The evidence of this theorem can be reduced to the following Lemma:

Lemma 1: The intersection of the fuzzy sets $S_{kw_i,d}$ and $S_{kw,k}$ for kw_i and kw is not empty if and only if $ed(kw,kw_i) \le k$.

Evidence: First, we show that $S_{kw_i,d} \cap S_{kw,k}$ is not empty when $ed(kw,kw_i) \le k$. To prove this, it is enough to find an element in $S_{kw_i,d} \cap S_{kw,k}$. Let $kw = e_1e_2 \cdots e_s$ and $kw_i = c_1c_2 \cdots c_t$, where all these e_i and c_j are single characters. After $ed(kw,kw_i)$ edit operations, kw can be changed to kw_i according to the definition of edit distance. Let $kw_* = e_{*1}e_{*2} \cdots e_{*m}$, where $e_{*i} = e_j$ or $e_{*i} = e_*$ if any operation is performed at this position. Since the edit operation is inverted, from kw_i, the same positions containing wildcard at kw_* will be performed. Because $ed(kw,kw_i) \le k$, kw_* is included in both $S_{kw_i,d}$ and $S_{kw,k}$, we get the result that $S_{kw_i,d} \cap S_{kw,k}$ is not empty.

Next, we prove that $S_{kw_i,d} \cap S_{kw,k}$ is empty if $ed(kw,kw_i) > k$. The proof is given by reduction. Assume there exists an kw_* belonging to $S_{kw_i,d} \cap S_{kw,k}$. We will show that $ed(kw,kw_i) \le k$, which reaches a contradiction. First, from the assumption that $kw_* \in S_{kw_i,d} \cap S_{kw,k}$, we can get the number of wildcard in w_*, which is denoted by n_*, is not greater than k. Next, we prove that $ed(kw,kw_i) \le n_*$. We will prove the inequality with induction method. First, we prove it holds when $n_* = 1$. There are nine cases be considered: If kw_* is derived from the deletion of both kw_i and kw, then, $ed(kw_i,kw) \le 1$ because the other characters are the same except the character at the same position. If the operation is deletion from kw_i and substitution from kw, we have $ed(kw_i,kw) \le 1$ because they will be the same after at most one substitution from kw_i. The other cases can be analyzed in a similar way and are omitted. Now, assuming that it holds when $n_* = \gamma$, we need to prove it also holds when $n_* = \gamma + 1$. If $kw_* = e_{*1}e_{*2} \cdots e_{*n} \in S_{kw_i,d} \cap S_{kw,k}$, where $e_{*i} = e_j$ or $e_{*i} = e_*$.

For a wildcard at position t, cancel the primary operations and relapse it to the unique typeset in kw_i and kw at this position. Assume two rudiments kw_{*i} and kw_* are resultant from them correspondingly. Then perform one operation at position t of tw_{*i} to make the character of tw_i at this position be the same with kw, which is denoted by kw'_i. After this operation, kw_{*i} will be changed to kw_*, which has only k wildcards. Therefore, we have $ed(kw'_i,kw) \le \gamma$ from the assumption.We know that $ed(kw'_i,kw) \le \gamma$ and $ed(kw'_i,kw_i) = 1$, based on which we know that $ed(kw_i,kw) \le \gamma + 1$. Thus, we can get $ed(kw,kw_i) \le n_*$. It renders the contradiction $ed(kw,kw_i) \le k$ because $n_* \le k$. Therefore, $S_{kw_i,d} \cap S_{kw,k}$ is empty if $ed(kw,kw_i) > k$.

Theorem 2: The EKST is secure concerning the search privacy.

Evidence: we need index privacy by using the reduction because in the wildcard based method the estimation of index and request of the same keyword is alike. We build an algorithm A' that utilizes A to determine whether some function $pf'(\cdot)$ is a pseudo-random function such that $pf'(\cdot)$ is equal to $pf(sk, \cdot)$ or a random function. A' has an access to an oracle $O_{pf'(\cdot)}$ that takes as input secret value x and returns $pf'(x)$. Which can be relaxed by adding some redundant trapdoors? A' picks one random $rb \in \{0, 1\}$ and sends kw_{*b} to the challenger. Then, A' is given a challenge value y, which is either computed from a pseudo-random function $pf(sk, \cdot)$ or a random function. A' sends y back to A, who answers with $rb' \in \{0, 1\}$. Suppose A guesses rb correctly with nonnegligible probability, which indicates that the value is not randomly computed. Then, A' makes a decision that $pf'(\cdot)$ is a pseudo-random function. As a result, based on the assumption of the indistinguishability of the pseudo-random function from some real random function, A at most guesses b correctly with approximate probability $1/2$. Thus, the search privacy is obtained.

VII. Conclusion

In this paper, we proposed EKST an efficient keyword searching technique for searching the fuzzy keyword over encrypted cipharcloud data while maintaining keyword confidentiality. The proposed searching technique deeply enhances system usability by presentencing the matching files when user's searching inputs exactly match the predefined keywords or the closest possible matching files based on keyword comparison semantics, when *exact* match fails. Our EKST greatly reduces the storage and depiction overheads, we develop revise space to enumerate fuzzified keywords similarity and implement a method on constructing fuzzified keyword sets, through accurate security study, we shown that our proposal is secure and privacy-preserving.

References

[1] Google, "Britney spears spelling correction," Referenced online at http: //www.google.com/jobs/britney.html, June 2009.
[2] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in *Proceedings of Crypto 2007, volume 4622 of LNCS*. Springer-Verlag, 2007.
[3] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. of IEEE Symposium on Security and Privacy'00*, 2000.
[4] E.-J. Goh, "Secure indexes," Cryptology ePrint Archive, Report 2003/216 2003, http://eprint.iacr.org/.
[5] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. of EUROCRYP'04*, 2004.
[6] B. Waters, D. Balfanz, G. Durfee, and D. Smetters, "Building an encrypted and searchable audit log," in *Proc. of 11th Annual Network and Distributed System*, 2004.
[7] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proc. of ACNS'05*, 2005. [8] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in *Proc. of ACM CCS'06*, 2006.
[9] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. of TCC'07*, 2007, pp. 535–554.
[10] F. Bao, R. Deng, X. Ding, and Y. Yang, "Private query on encrypted data in multi-user settings," in *Proc. of ISPEC'08*, 2008.
[11] C. Li, J. Lu, and Y. Lu, "Efficient merging and filtering algorithms for approximate string searches," in *Proc. of ICDE'08*, 2008.
[12] A. Behm, S. Ji, C. Li, , and J. Lu, "Space-constrained gram-based indexing for efficient approximate string search," in *Proc. of ICDE'09*.
[13] S. Ji, G. Li, C. Li, and J. Feng, "Efficient interactive fuzzy keyword search," in *Proc. of WWW'09*, 2009.
[14] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. N. Wright, "Secure multiparty computation of approximations," in *Proc. Of ICALP'01*.
[15] R. Ostrovsky, "Software protection and simulations on oblivious rams," Ph.D dissertation, Massachusetts Institute of Technology, 1992.
[16] V. Levenshtein, "Binary codes capable of correcting spurious insertions
and deletions of ones," *Problems of Information Transmission*, vol. 1, no. 1, pp. 8–17, 1965.