Achieving Privacy-Preserving Edit Distance Query in Cloud and Its Application on Genomic Data

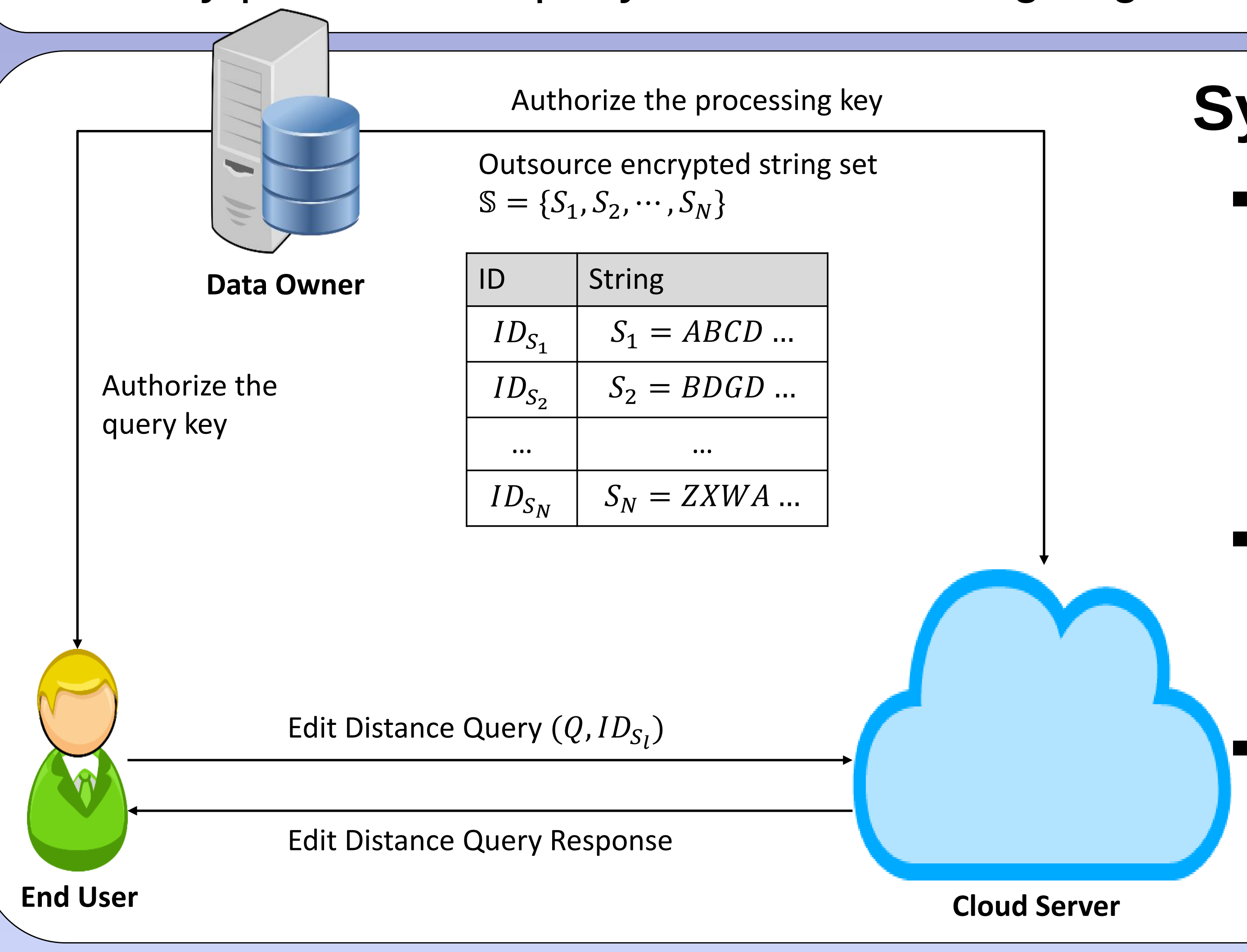
Jason Chang (Undergraduate) and Rongxing Lu

Contact Email: jchang@unb.ca, rlu1@unb.ca

Canadian Institute for Cybersecurity (CIC), Faculty of Computer Science, University of New Brunswick (UNB)

ABSTRACT

Edit distance is one of the most frequently used metrics for sequence comparison in many fields, including computational biology. Edit distance approximates the similarity between sequences by counting the minimum number of operations required to transform one sequence into the other. With recent advances in cloud computing, cloud servers are capable of providing large-scale storage and powerful computational resources, which motivates researchers to outsource their data for edit distance computation on larger data sets. However, outsourcing data and computing edit distance on the cloud raise concerns over privacy. In this paper, we propose a secure model for achieving privacy-preserving edit distance query in cloud. The proposed scheme combines BGN encryption and improved data generation method to securely and efficiently process the query without revealing original data to the cloud. Additionally, we explore applications on genomic data with block-wise edit distance approximation.


Authorize the processing key
Outsource encrypted string set
$\mathbb{S} = \{S_1, S_2, \cdots, S_N\}$
Data Owner
Authorize the query key
Edit Distance Query (Q, ID_{S_l})
Edit Distance Query Response
End User
Cloud Server

ID	String
ID_{S_1}	$S_1 = ABCD\ldots$
ID_{S_2}	$S_2 = BDGD\ldots$
…	…
ID_{S_N}	$S_N = ZXWA\ldots$

System Model

- **Data Owner:** A service provider owns a set of strings $S = \{S_1, S_2, S_3, \cdots, S_N\}$ which is encrypted before outsourced to the cloud server. Data owner will also authorize the end user with some query key. Data owner will not participate in the concrete edit distance query.
- **Cloud Server:** A powerful entity that stores the outsourced strings $S = \{S_1, S_2, S_3, \cdots, S_N\}$ from the data owner and processes the edit distance queries from the authorized end user.
- **End User:** An authorized user that owns the query key and will query those records of interest to him/her. It will encrypt its query string before launching a query.

1. BGN Key Authorization

- Given the security parameter κ, the data, owner first generates the bilinear parameters (N, g, G, G_T, e) by running $CGen(\kappa)$, where $N = pg$. Then, the data owner sets the BGN public key $pk = (N, G, G_T, e, g, h)$ and the private key $sk = p$, where $h = g^q$. Further, the data owner chooses a cryptographic hash function $H : \{0.1\}^* \rightarrow Z_N$, and also chooses a secret key $\kappa \in Z_N$. Finally, the data owner keeps (p, κ) secretly, and publishes $pk = (N, G, G_T, e, g, h)$ and $H : \{0.1\}^* \rightarrow Z_N$

2. Improved BGN Encrypted Data Generation

- The data owner, given a query key k, will use BGN encryption to generate ciphertext $[S_i]$ for each letter S_i *in a string* S, $[S_i] = E_T(ID_S, k, S_i) = e(g,g)^{p \cdot H(ID_S||k||S_i)} \in G_T$. An end user will prepare its string Q, $[Q_j] = E_T(ID_S, K, Q_j, r_j) = g^{H(ID_S||k||Q_j)} \cdot h^{r_j} \in G$, with a random number $r_j \in Z_N$.
- However, when two characters S_i, S_j *in a string* S are the same, we also have $[S_i] == [S_j]$. Therefore, instead of repeating the BGN encryption operation, we can perform the improved data generation:

1. First, randomly order all letters in a set of alphabetical letters $M = \{'A', 'B', \cdots, 'Z'\}$.
2. We generate the ciphertext $[M_j]$ for each letter $M_j \in M$ and set $[M_j].ID$ = Ord(M_j).
3. When a character S_i is equal to M_j, we can simply set $[S_i].ID = Ord(M_i)$ and obtain $[S_i]$ by reading $B_l[[S_i].ID]$.

Ord	0	1	2	3	4	5	6	7	8	9	10	11	12
M_j	‘M’	‘I’	‘D’	‘L’	‘F’	‘N’	‘A’	‘S’	‘E’	‘O’	‘T’	‘U’	‘Z’
Ord	13	14	15	16	17	18	19	20	21	22	23	24	25
M_j	‘X’	‘I’	‘B’	‘V’	‘G’	‘K’	‘W’	‘C’	‘H’	‘Y’	‘P’	‘Q’	‘R’

The array to store randomly ordered set $M = \{'A,''B', \cdots, 'Z'\}$.

$\longrightarrow$

ID	0	1	2	3	⋯	22	23	24	25
$[M_j]$	[‘M’]	[‘I’]	[‘D’]	[‘L’]	⋯	[‘Y’]	[‘P’]	[‘Q’]	[‘R’]

The array B_l to store (index = $[M_j]$.ID, value = $[M_j]$) for all $M_j \in M$.

3. Edit Distance Computation

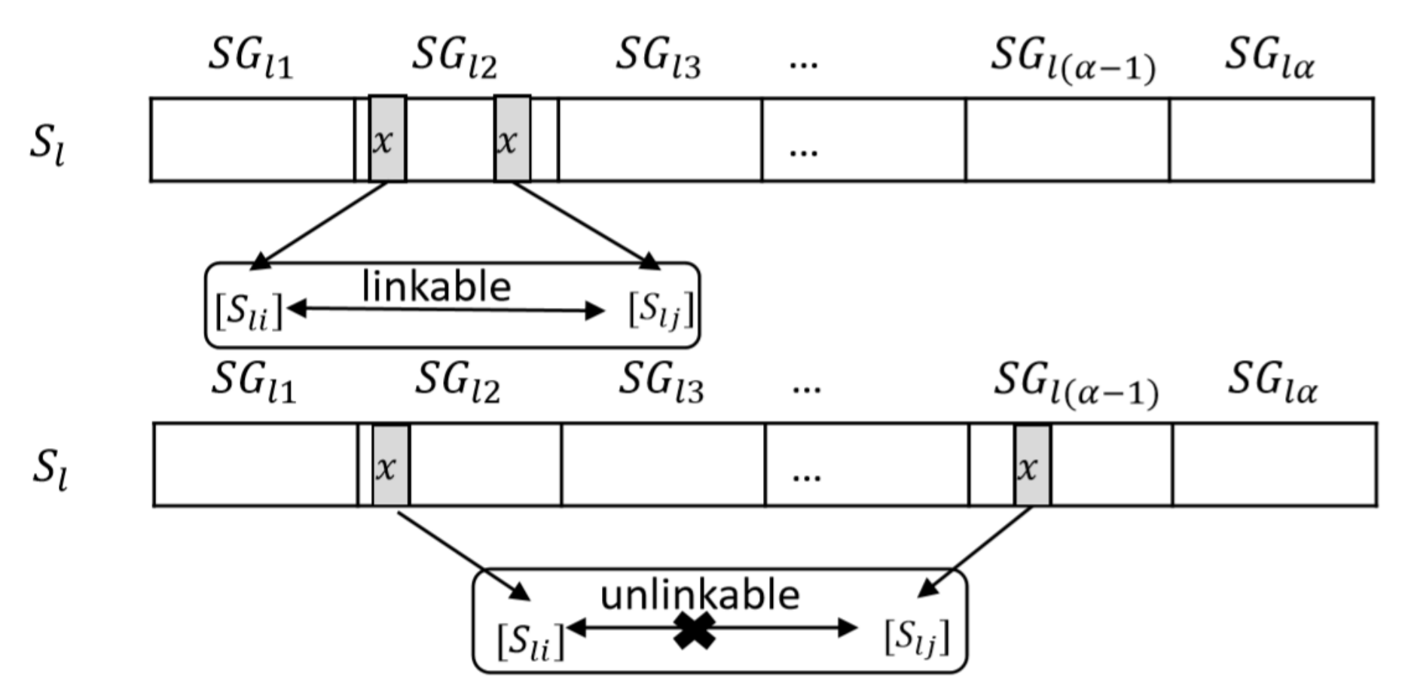
```
Algorithm 5: PRIVACY-PRESERVING EDIT DISTANCE
Input: processing key $g^{pt}$ and $([Q], [S_l].ID, B_l)$
Output: the edit distance $ED(S_l, Q)$
1  compute $n = length([S_l].ID)$, $m = length([Q])$;
2  initialize an edit distance matrix $(A_{i,j})_{0\le i\le n, 0\le j\le m}$;
3  for $i$ = 0 to $n$ do
4      $A_{i,0} = i$;
5  for $j$ = 0 to $m$ do
6      $A_{0,j} = j$;
7  for $i$ = 1 to $n$ do
8      for $j$ = 1 to $m$ do
9          compute $[[Q_j]] = e([Q_j], g^p) = e(g,g)^{p\cdot H(ID_{S_l}||k||Q_j)}$;
10         if $B_l[[S_{li}].ID] == [[Q_j]]$ then
11             $A_{i,j} = A_{i-1,j-1}$;
12         else
13             $A_{i,j} = 1 + \min(A_{i-1,j}, A_{i,j-1}, A_{i-1,j-1})$;
14 return $A_{n,m}$; // the edit distance $ED(S_l, Q) = A_{n,m}$
```

Note that, without knowing the key k, the cloud cannot recover S_i, Q_j from$e(g,g)^{p \cdot H(ID_S||k||S_i)}$, $e(g,g)^{p \cdot H(ID_S||k||Q_j)}$, because the unknown k makes it impossible to recover them by brute-force guessing. On the other hand, since the data owner knows k, he can recover Q_j by brute-force guessing each value x in space, compute $H(ID_S||k||x)$, and compare whether $e(g,g)^{p \cdot H(ID_S||k||Q_j)} == e(g,g)^{p \cdot H(ID_S||k||x)}$. If yes, the data owner recovers the correct Q_j.

Application on Genomic Data

- In our proposed scheme, the probability to correctly guess S_i from $[S_i]$ is $\frac{1}{26!}$. However, there are only 4 characters for set $M = \{'A', 'C', 'G', 'T'\}$ in genomic data and thus a higher probability $\frac{1}{4!}$. To address this challenge, we apply the segmentation technique.
- We divide each string $S_l \in S$ into α segments $S_l = \{SG_{l1}, SG_{l2}, \cdots, SG_{l\alpha}\}$ and include the identifier $ID_{SG_{lk}}$ for each segment $SG_{lk} \in S_l$ into the encryption. In such a way, when two characters S_{li} *and* S_{lj} are in the same segment of a string S_l, we can link their ciphertexts. However, if S_{li} *and* S_{lj} are in different segments of a string S_l, we cannot link their ciphertexts. With this strategy, we can greatly improve the security of each string $S_l \in S$ to $\frac{1}{(4!)^{\alpha}} = \frac{1}{24^{\alpha}}$.
- In addition, previous studies have reported block-wise approximation algorithm for the edit distance function. Therefore, we can compute approximate edit distance query defined as:

$$ApproxED(S_l, Q) \approx \sum_{k=1}^{\alpha} ED(SG_{lk}, Q_{SG_k})$$



```
Algorithm 8: PRIVACY-PRESERVING EDIT DISTANCE FOR GENOMIC DATA
Input: processing key $g^{pt}$ and
       $([Q] = \{[Q_{SG1}], [Q_{SG2}], \cdots, [Q_{SG\alpha}]\}, \{B_{l1}, \cdots, B_{l\alpha}),$
       $\{[SG_{l1}].ID, \cdots, [SG_{l\alpha}].ID\})$
Output: the approximate edit distance $ApproxED(S_l, Q)$
1  for $k$ = 1 to $\alpha$ do
2      compute $n = length([SG_{lk}].ID)$, $m = length([Q_{SGk}])$;
3      initialize an edit distance matrix $(A_{i,j})_{0\le i\le n, 0\le j\le m}$;
4      for $i$ = 0 to $n$ do
5          $A_{i,0} = i$;
6      for $j$ = 0 to $m$ do
7          $A_{0,j} = j$;
8      for $i$ = 1 to $n$ do
9          for $j$ = 1 to $m$ do
10             compute $[[Q_{SGkj}]] = e([Q_{SGkj}], g^p) =$
               $e(g,g)^{p\cdot H(ID_{S_l}||ID_{SG_{lk}}||\kappa||Q_{SGkj})}$;
11             if $B_{lk}[[SG_{lki}].ID] == [[Q_{SGkj}]]$ then
12                 $A_{i,j} = A_{i-1,j-1}$;
13             else
14                 $A_{i,j} = 1 + \min(A_{i-1,j}, A_{i,j-1}, A_{i-1,j-1})$;
15     set $ED(SG_{lk}, Q_{SGk}) = A_{n,m}$; // the edit distance
16 return $ApproxED(S_l, Q) \approx \sum_{k=1}^{\alpha} ED(SG_{lk}, Q_{SGk})$;
```