ISSN NO. 2456-3129

IJEPAS

International Journal of Engineering, Pure and Applied Sciences
Vol. 3, No. 1, March-2018

HOMOMORPHIC ENCRYPTION FOR MEDICAL ANALYSIS ON GENOME SEQUENCE

Krishan Gupta[1], Shubham Sinha[2], Deepanshu Jain[3]
Bharati Vidyapeeth's College of Engineering, New Delhi[1,2,3,4]
Email: *Krishanc3@gmail.com[1], deepakshu48jain@gmail.com[2], shubhamsinha125@gmail.com[3]*

Abstract-Homomorphic encryption addresses the problem of security breach of trusted parties by conversion to ciphers which maintain the arithmetic relations between them as in the message space. Any private computation outsourcing thus gets more secured as the vendor can compute on encrypted values without knowing the meaning of data. Here we present the encryption scheme for genome analysis for deriving medical results. We have used the Paillier's algorithm which is homomorphic under addition and is sufficient for matching and aggregating events. We show some types of searches performed for various tests on genome sequence and will generalize the search function so that new search patterns can be accommodated just by changing parameters to the unified function.

Keywords- .Homomorphic , Paillier, Gemone, algorithm, function

1. INTRODUCTION

1.1. *Homomorphic Encryption*

Homomorphic encryption is a cryptographic scheme that allows mathematical operations on data to be carried out on cipher text, instead of on the actual data itself. The cipher text is an encrypted version of the input data (also called plain text). It is operated on and then decrypted to obtain the desired output. The critical property of homomorphic encryption is that the same output should be obtained from decrypted the operated cipher text as from simply operating on the initial plain text.

The process begins with some plain text message, *m.* The goal is to perform some function *f* on it. It would be safer to encrypt the message using enc before performing any functions on it. So the message is encrypted to some cipher text 163726.

Then, it is evaluated, or transformed, into another value using some other function, *f=x+127,* for example. The output, 163853, is another completely encrypted message. This message can then be decrypted.

$$\begin{tikzcd}
\text{enc(m)} \arrow[r, "\text{evaluate}"] & \text{enc(f(m))} \arrow[d, "\text{decrypt}"] \\
\text{m} \arrow[u, "\text{encrypt}"] \arrow[r, "\text{function}"] & \text{f(m)}
\end{tikzcd}$$

<u>Fig. 1. Homomorphism and encryption</u>

Homomorphic encryption is of 3 types:
1. Partially homomorphic encryption
2. Somewhat homomorphic encryption
3. Fully homomorphic encryption

1.2. Paillier Algorithm

1. Key generation: Let p and q be prime numbers where p < q and p does not divide q − 1. For the paillier encryption scheme, we set the public key pk to n where n = pq and private key pr to (λ, n) where λ is the lowest common multiplier of p − 1, q − 1.
2. Encryption with the public key: Given n, the message m, and a random number r from 1 to n − 1, encryption of the message m is calculated as follows:

.

$$E_{pk}\,(m) = (1 + n)^m r^n \bmod n2$$

3. Decryption with the private key: Given n, the cipher text c = E_{pk} (m), we calculate the D_{pr} (c) as follows: m = [((cλ mod n2) − 1)/n]λ−1 mod n where λ−1 is the inverse of λ in modulo n.
4. Adding two ciphertexts (+h): Given the encryption of m1 and m2 , E_{pk} (m1) and Epk (m2), we calculate the Epk (m1 + m2) as follows:

$$\begin{aligned}
&E_{pk}\,(m1\,)E_{pk}\,(m2\,) \bmod n^2 = ((1+n)^{m1}\\
&r^n{}_1\,)\,((1 + n)^{m2}\, r^n{}_2\,) \bmod n^2\\
&=((1 + n)^{m1\,+m2}\,(r_1 r_2\,)^n) \bmod n^2\\
&= E_{pk}\,(m1 + m2\,).
\end{aligned}$$

Note, due to the modular operation, cipher text addition yields Epk (m1 + m2 mod n).
5. Multiplying a ciphertext with a constant ($\times_h$): Given a constant k and the encryption of m1, Epk (m1), we calculate k ×h Epk (m1) as follows:

$$k \times h \text{ Epk } (m1) = \text{Epk } (m1)^{k}$$
$$\text{mod } n^{2} = ((1 + n)^{m1}\ r^{n1}\ )^{k}$$
$$\text{mod } n^{2}= (1+n)^{km1}\ r_{1}^{kn} \text{ mod} n^{2}$$
$$= E_{pk}\ (km1\ ).$$

It is the DNA sequence containing the nucleotides represented by alphabets (A,T,G,C). There are 22 chromosome pairs (44 individual chromosomes each having 2 strands) and 2 sex chromosomes (XX for female, XY for male). The DNA sequence is representative of various traits of an individual represented by genes. Also various anomalies in this sequence is indicative of a disease or reveal susceptibility to a disease. The DNA mapping was fully studied by the 'Human Genome Project'.

2. SOME APPLICATIONS OF MEDICAL ANALYSIS FROM GENOME SEQUENCE

1. Diseases arising from triplet repeat expansion: Occurrence of a triplet pattern above the normal limit. Example: Huntington disease –occurrence of pattern 'CAG' more than 35 times is indicative of the disease.
2. Organ transplant: Checking compatibility of patient and donor by gene matching in particular regions. HLA is a gene which distinguishes self-cells from foreign cells, responsible for tissue rejection. Genes for HLA should be as similar as possible. 6 HLA genes need to be compared at location 6p21.33/32.
3. DiGeorge syndrome: Deletion in 22q11.21
4. Sickle cell anemia: If both parents have 'CAC' pattern instead of 'CTC' pattern in one of the strand in 11p15.5, then the child has 25% chance of having 'CAC' in both strands. Sickle blood cells have less oxygen carrying capacity and the patient have much reduced lifespan.
5. Chronic Myelogenouslukemia: Search chromosome 22 for shortening of length. Look for 'BCR-ABL' gene which is ontogeny.

3. METHEDOLOGY

1. Genome sequence of 46 chromosomes against a person is stored in DB.
2. Each sequence is a multiple of 3 because functional parts of DNA are always looked up as combination of 3 nucleotides.
3. The constituent letters (A,T,G,C) can be given codes as (1,2,3,4) respectively. Thus a triplet code is 444 at max.
4. Such 3 digit numbers can be separately encrypted and concatenated.
5. String matching can be done by the homomorphic subtraction operation. A match of search triplet will yield 0s on decrypting the result cipher text.

4. COMMON SEARCH ALGORITHM

Writing a function for each disease lookup will be time and space consuming. New important tests can come up anytime and we don't want site maintenance for each of such events. Therefore we intend on implementing a single unified function which can be modified for any special search. Thus, users will have to pass only the search parameters to invoke the function at server. Generalising the search pattern from the above cases we can include the following parameters:

1. Search at specific locations or the entire genome sequence.
2. Enable count length option for the given search area to check for increased or decreased length.
3. Option to search both chromosomes of a given number. Usually they are same but can differ slightly due to modifications.

5. FAST SEARCH ARCHITECTURE:

Abiding by semantic security principle, we should ensure that the search request shouldn't give any information about the search query such as which results from the database were accessed corresponding to a given search term. So, in worst case, all records of the database have to be processed. We therefore propose a 3-server architecture to speed up the process. The search request will be composed of 3 parts: patient id, location, search term.

Patient id	Location(s)	Search term(s)

The 3 server architecture is as follows. We assume that the servers are independently managed i.e. they do not share the private keys among them. Server 1 contain the patient ids, Server 2 is only processing server which decrypts information from server 1, Server 3 contains the genome data of patients and is in one to one relation with server 1's patient id table record.

Paillier is a public-key cryptosystem. Encryption is done at client: Patient's id with public key whose private key is stored on server 2 so that it reveals the entry to be processed, location with public key whose private key is stored on server 3 so that the column to be processed is revealed, search term with key whose private key is with the client itself so that client data is never exposed. Server 3 can send the processed string to client which will decrypt it and infer from the occurrences of 0's.

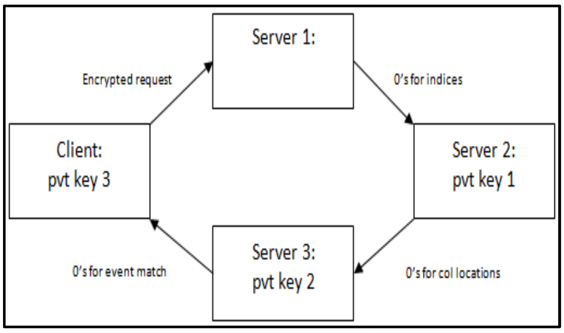

Fig. 2. Three server architecture

6. CONCLUSION:

The paillier encryption scheme is efficient for searching. We only have to understand the fact that decrypting the product of 2 ciphers gives their sum and with little modification (multiplicative inverse of 2nd term) gives difference. Decrypting the processed string and searching for 0's will give the occurrences of pattern match.

However, semantic security is an issue and to protect it one might have to search the entire table for a single query. Assuming a multi-level server architecture and breaking the algorithm to each of them such that the part assigned can only be solved by them, we can still maintain semantic security to a large extent provided that the servers never share their private keys. A malicious user trying to make meaning of the search query has to know all the private keys distributed in the system. Also, it improves the execution time as we can now directly go to the search location of genome string without worrying about violation of semantic security.

REFERENCES

[1] Wenjie Lu, Yoshiji Yamada, Jun Sakuma: (2015) "Efficient Secure Outsourcing of Genomewide Association Studies", IEEE CS Security and Privacy Workshops.

[2] Alex Page, OvuncKocabas, Scott Ames ,MuthuramakrishnanVenkitasubramaniam, TolgaSoyata: (2014) "Cloud□based Secure Health Monitoring: Optimizing Fully□Homomorphic Encryption for Streaming Algorithms", Workshop □ Cloud Computing Systems, Networks and Applications, Globecom.

[3] Sophia Yakoubov, Vijay Gadepally, Nabil Schear, Emily Shen, Arkady Yerukhimovich: (2015) "A Survey of Cryptographic Approaches to Securing Big□Data Analytics in the Cloud", high Performance Extreme Computing Conference (HPEC), IEEE.

[4] Jung HeeCheon, Miran Kim, Kristin Lauter: (2017) "Homomorphic Computation of Edit Distance", Seoul National University (SNU), Republic of Korea.

[5] Mete Akgün, A. Osman Bayrak, Bug˘raÖzer, M. SamilSag˘ırog˘lu: (2015) "Privacy preserving processing of genomic data: A survey", Journal of Biomedical Informatics.

[6] Nathan Dowlin, Ran Gilad Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, John Wernsing; (2017) "Manual for Using Homomorphic Encryption for Bioinformatics", IEEE.