POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

A Fully-Homomorphic Encryption System for Privacy-Preserving Network-Based Contact Tracing

Laurea Magistrale in Telecommunications Engineering - Ingegneria delle Telecomunicazioni

Author: Paul Horvath-Bojan

Advisor: Prof. Massimo Tornatore

Co-advisors: Dr. Davide Andreoletti,Dr. Omran Ayoub

Academic year: 2021–2022

1. Introduction

The present thesis presents a network-based contact tracing protocol using fully-homomorphic encryption (FHE), proposing a privacy-preserving approach to passively assess risk of individuals in the case of a contagion such as Covid-19.

The need for a network-based privacy-preserving contact tracing protocol is given by the lack of a sufficiently widespread alternative. The status quo is represented by contact tracing solutions involving smartphone applications leveraging either user proximity to one another or user locations. The lack of adoption is justified by either the lack of tech-savvy or distrust towards the apps — either due to potential vulnerabilities, battery consumption, or unwillingness to share personal data. An existing network-based approach [1] addresses the app-related issues. It employs two traditional encryption schemes (Pailler asymmetric encryption coupled with Shamir secret sharing) to provide a higher-overhead implementation.

The protocol proposed in this work instead uses FHE to encrypt sensitive location and disease infection status data regarding individuals and performs private calculations to assess the contact between users and their risk of infection as a function of proximity to other infected individuals. Figure 1 presents a sketch of protocol functionality.

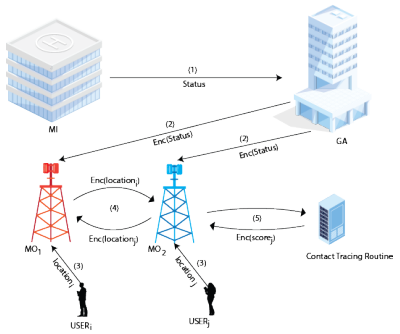

Figure 1: Sketch of protocol functionality.

In this study, we formally state a novel problem for assessing the risk of infection of individuals by tracing their contact with confirmed positive cases of contagion in a network-based environment using homomorphic privacy-preserving primitives.

This paper is organized as follows. We first introduce the theoretical background needed to understand the construction of the proto-

col, followed by a description of the system architecture, the formal problem statement, the privacy requirement considered, and the implementation aspects of the protocol. Not lastly, we discuss the simulation setup used, as well as the results obtained, before drawing conclusions and establishing new directions for future work.

2. Theoretical background

Fully-homomorphic encryption (FHE) is an encryption scheme that enables analytical functions to be run directly on encrypted data while yielding the same encrypted results as if the functions were run on plaintext. It was first theorized by Rivest, Adleman and Dertouzos in [5], with a first implementation by Gentry in [4]. In terms of Boolean (AND/OR) or arithmetic (addition/multiplication) circuit support, homomorphic encryption can be classified as:

- Partial (PHE) — only allow one type of gates (e.g., addition or multiplication);
- Somewhat (SHE) — can evaluate two types of gates on limited number of circuits;
- Levelled (LHE) — can evaluate arbitrary circuits of limited depth;
- Full (FHE) — can evaluate arbitrary circuits of arbitrary depth.

All modern HE approaches encrypt with noise, which grows as more operations are performed. Until a certain noise bound, the ciphertexts can still be successfully decrypted; beyond that, the information is lost. The noise growth during additions is negligible — noise size of sum equal to max noise size in terms — and hence essentially an infinite number of additions can be performed. The noise growth during multiplications, however, is significant — noise size of product equal to sum of noise sizes in factors — and hence only a limited amount of multiplications can be performed before consuming the noise budget. Noise is reset and ciphertexts re-encrypted under a new key via a procedure called bootstrapping. However, bootstrapping is the most computationally costly and time-consuming FHE operation, and incurs additional overhead for the new encryption keys so it was avoided in this work. The proposed protocol was thus adapted to require only a fixed amount of multiplications and hence an LHE approach was taken, using the Cheon-Kim-Kim-Song (CKKS)[2] scheme as support.

2.1. CKKS

Within this scheme, vectors of Gausssian numbers — i.e., numbers belonging to the set $Z[i] = \{a + bi \ : \ a, b \in Z\}$ — can be encrypted, with computations being performed in a single instruction–multiple data fashion. This particular scheme was chosen due to its particularity of being able to perform computations on floating point encrypted numbers.
CKKS batches multiple (N) plain complex values into a single CKKS vector plaintext. This vector is then encoded by embedding into the a high-degree ($2N$) integer polynomial field with coefficients bounded by a ciphertext modulus c_mod — with elements represented as vectors of coefficients — after which all coefficients are scaled by a high scaling factor Δ. The used secret keys are vectors of the same length as the ciphertext vectors, with a quarter ± 1 values and the rest 0. The public keys are pairs of $2N$ vectors/polyonmials, one which retains knowledge of the secret key behind a random linear combination, with a second being just random. Encryption returns ciphertexts consisting of two vectors — a linear combination of the public key and the encoding of the plaintext. The scheme allows essentially an unlimited number of additions, and subsequent multiplications up to the established maximum level (hard bound $\lfloor \frac{c\_mod}{\Delta} \rfloor$) between ciphertexts. Multiplication essentially transforms two-dimensional ciphertexts into a three-dimensional result. In order to pass back to two dimensions and continue operations, a relinearization process has to be performed, which involves a costly high-modulus operation. A particular functionality of CKKS is the possibility to rescale ciphertexts by dividing both the ciphertext and the its coefficient modulus by a division factor. This reduces the size of the ciphertext, allowing for quicker multiplications. Decryption is performed by a linear combination with the secret key. In general, it is not advised to further share results of decryption to an untrusted party.

3. System architecture

Within this section, a high level description of the protocol proposed in this work is given. The architecture is inspired by the work done in [1], with a few modifications. The protocol involves a number of mobile operators (MO), along with a government agency (GA) and the users identified with their devices. The MOs are assumed to have localization capabilities of sub-metre accuracy for their users, and to have access to the latest available locations. The GAs are assumed to know the infection status of all users. The users are identified with their mobile devices, and each provides a public encryption key to their respective MOs, as well as the GA. From the user side, the protocol appears to be almost fully passive. The GAs sends encrypted values of the users' infection status to the respective users' MO with relatively low frequency (e.g., daily). The MOs exchange encrypted values of their users' location data with relatively high frequency (e.g., every minute, every 30 seconds). On the received encrypted data, the MOs perform homomorphic operations to obtain an encrypted contact assessment score. Afterwards, they perform a final homomorphic multiplication between the contact assessment score and the encrypted infection status in cipher domain to assess the instant score. The final score of each user is the sum of all the instant scores up to the considered point in time.
At any moment, the users can poll their MO to receive the encrypted score, which only they can decrypt.

4. Problem statement

The scope of this protocol is computing a risk score for each user, given their locations and infection statuses.
Let δ be the threshold distance below which two users are in contact. Then two users are in contact at a certain time t if the distance between them is smaller than δ. More formally:

$$contact_{ij}^{(t)} = \begin{cases} 1, \text{ if } Dist(User_i, User_j) \leq \delta, \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

The risk score of a user at time t_0 is calculated as:

$$score_i^{(t_0)} = \sum_{t \leq t_0} \sum_{j:contact_{ij}^{(t)}=1} status_j^{(t)}. \tag{2}$$

5. Privacy Requirements

The considered privacy goals were:

1. Location confidentiality. A User's location is only known to the user itself and their MOs.
2. Contact confidentiality. Information about whether a User is in contact with another User and the number of times they were in contact can only be known to the respective Users themselves. If the Users share the same MO, then the MO may be a party to this information; if the Users are from different MOs, then any MO cannot know the identity of the other MO's User, nor can they know the number of times they were in contact.
3. Infection status and score confidentiality. User infection status can only be known to the User themselves and the GA. A User's risk assessment score is by default only known to themselves.

The considered attacker model is an honest-but-curious attacker — i.e., the MOs and the GA will execute the protocol truthfully, but given the opportunity, will try to learn as much as possible about the users.

6. Implementation

6.1. Area of Effect Tessellation

To keep the implementation practical, a tessellation (i.e., splitting into sub-surfaces) of the protocol area of effect is considered, so as not to have the system compare all the users' locations with one another. The tessellations are considered squares of equal sides l. To this end, each such tessellation area is assigned an area ID, and within the MO database, each user has an associated area ID. The area IDs are exposed also when communication between MOs takes place, and hence the search for potential contact is only performed within the area assigned to the user, as well as the 8, 5, or 3 adjacent areas — the 5 areas are considered for edge cases, while the 3 are considered for corner cases.

6.2. Approximate Contact Formula

By construction, FHE does not directly provide support for comparisons between ciphertexts. Instead, a number of repeated additions and multiplications can be used to approximate a step function that assigns 1 to distances less than the threshold δ and 0 to greater distances. The chosen formula is:

$$contact_{ij} = (1 - \frac{Dist(User_i, User_j)}{8l^2})^{2^k}. \quad (3)$$

The exponent is 2^k because that is the maximum achievable exponent in a k-level setup for CKKS. The formula is a smooth approximation of the step function at δ if the parameters are chosen appropriately. We report in figure 2 a graph comparing the step function with the used approximation formula for $k = 11, 12, 13$.

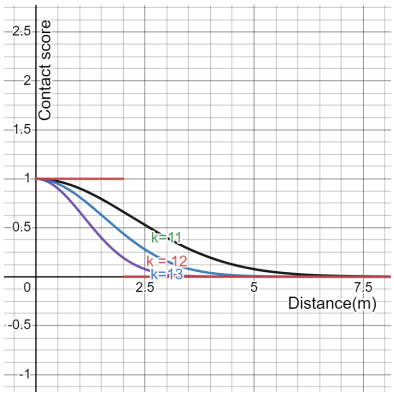

Figure 2: Approximation vs step for $\delta = 2, l = 50$.

6.3. CKKS Parameters

In terms of chosen parameters for CKKS, the results that come in support of setting parameters satisfying security are taken from [3]. Assessing which parameters to choose is done by evaluating, in order of priority:

1. Security level — as the data encrypted was sensitive, we opted for $\lambda - 128$bit. This directly affects the chosen Hamming weight of the secret key, as well as the pair of maximum plaintext size and ciphertext coefficient moduli;
2. Multiplication level — this depends on the considered exponent k in formula 3, as bootstrapping is avoided. In addition to the k levels needed by the formula, another two multiplications are done with ciphertexts — one when squaring to compute squared distance between users, and one with the status of the contact user;
3. Precision — as high as possible, in the conditions set by the higher priority constraints. Given the moduli of plaintext and ciphertext, choose the highest scaling factor that allows the needed multiplication level.

For the considered system,

7. Results

Having chosen the parameters for the encryption system — polynomial degree 256, ciphertext modulus 2^{744}, scaling factor 2^{49}, big modulus $2^{9}30$, as if having contact threshold 2 and tessellation area side length 50 — the average overhead was estimated for certain user population counts, sizes of tessellation, and total sizes of the protocol area of effect. MO overheads are reported as megabytes per user in figures 3 and 4.

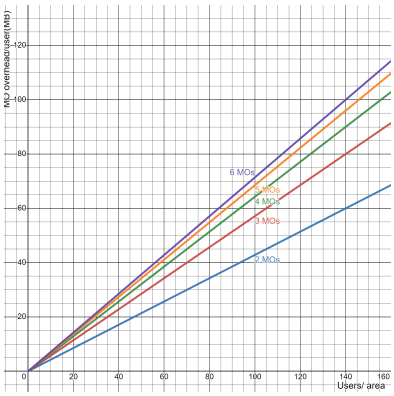

Figure 3: MO overhead per user as a function of users/tessellation.

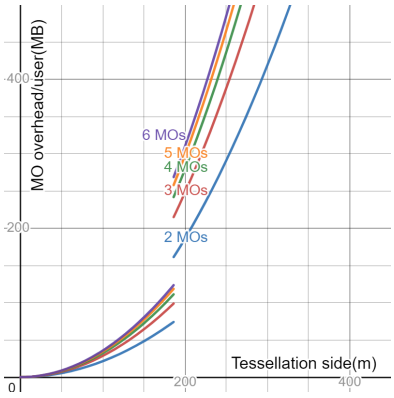

Figure 4: MO overhead per user as a function of tessellation side.

The MO overhead fluctuations are mainly accounted for by the number of contact evaluations performed for each user. Considering two participating MOs, average MO overhead per protocol round for a Milan-level density of 5G-capable users (considered $5000/km^2$) is 8.142MB/user, while considering Manila-level density of individuals, all with 5G capabilities ($43000/km^2$), the overhead can reach 77.138MB/user. With respect to tessellation area side, the registered overhead suffers a significant increase around tessellation side 186m, as the exponentiation factor needed for the approximation formula incurs a much higher ciphertext size — 2^{1618}. Figure 4 assumed a constant user density of $5000/km^2$. For two MOs at the lower bound of the 186m graph jump, MO overhead per user is 74.13MB, while at the upper bound it is 161.212MB.
GA overheads are much higher, but their communication phase also occurs much less frequently. Figures 5 and 6 report the overheads in gigabytes.

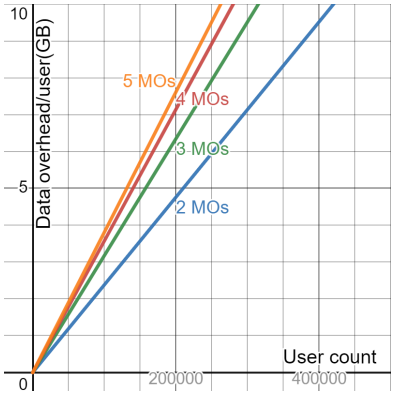

Figure 5: GA overhead per user as a function of total user count.

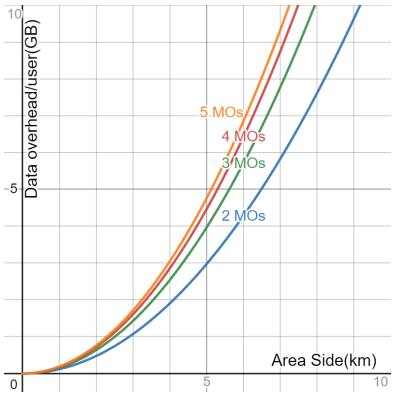

Figure 6: GA overhead per user as a function of area of effect side.

GA overhead reaches 2.381 GB per user each communication phase for 100000 users, considering 2 MOs running the protocol. Considering a density of 5000 users/km^2, GA overhead can reach up to 19 GB overhead per user on a $100km^2$ square area for 5 involved MOs.
A prototype Python program was built that simulated the behaviour of the protocol using the py-fhe library implementing CKKS. Two types of simulations were run:

- Run time assessment for single operations over ciphertexts, as used in the protocol;
- Running three variants of the protocol to compare score results, as well as register run times. The three variants were, respectively, a version using formula 1 exchanging information in plaintext, a version using formula 3 exchanging information in plaintext, and a version using formula 3 using the capabilities of CKKS.

On a personal computer, for the parameters chosen, a single multiplication of two fresh ciphertexts takes 1.63s, which translates to an average time to perform the contact approximation evaluation of 19.17s, while score evaluation will take a mean of 20.74s. Multiplications are the most costly operations in the scheme, and additionally they occur most frequently.
Protocol round run times were high, due to a mix of limited computational capabilities of the hardware used, single thread usage, as well as the nature of the encryption scheme. Per round, on a user population of 10, all within adjacent areas at all times, the reported average run time was 18 minutes.
Finally, we report the score correspondence between the variants of the protocol used in the simulations. The threshold score is considered the ideal value, and without additional errors from CKKS usage, the corresponding relation between using formula 1 and 3 is reported in figure 7.

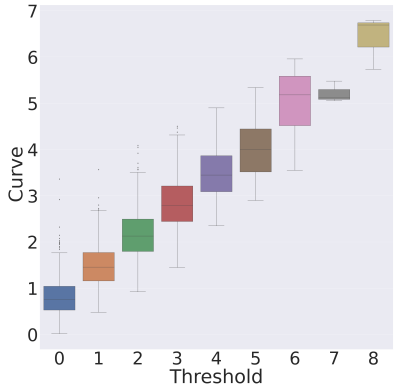

Figure 7: Relation between threshold score approximate score.

In figure 7, the considered threshold is $\delta = 2$, and the corresponding exponent is 2^{12}. It is the result of running the two plaintext variants of the simulation program over a population of 500 for 100 rounds over a 200m $\times$ 200m area, considering 50 infected. Lower contact thresholds achieve better separation of approximate scores with respect to the threshold-based ones at the cost of higher computational complexity due to the need for higher exponents, while higher contact thresholds present a more overlapped distribution of approximate scores while needing a lower exponent.
Not lastly, the errors in computation between the CKKS-using variant and the plaintext variant using the approximation present a worst-case absolute error of 0.6634, with a value of order 10^{-2} and a median of order 10^{-8}.
Results show that the protocol achieves low errors, and that the approximations made behave in a monotonous fashion similar to threshold approaches. Under parameters guaranteeing 128 bits of security, a linear growth with respect to user density of MO overhead per user is observed, with 5 MB per user for Milan-like density. Moreover, the GA overhead per user presents a similar behaviour, with comparable overhead with respect to the frequency of communication phases for 5 MOs with 189000 users (i.e., by dividing daily GA overhead to the number of MO communications in a day). The drawbacks are the sizeable computation costs to be paid, with run times of minutes for even low (10–20) user counts.

8. Conclusions and Future Work

To conclude, one can observe the benefits and the drawbacks of a network-based FHE approach to contact tracing. The accuracy of the employed approximations provides a good separation of scores, while ensuring a high security level for encrypting sensitive personal data. The communication overheads generated per user by the scheme are relatively low, yet scale with user count — quadratically for the MO, linearly for the GA.
As ideas for future work, we suggest exploring additional implementations, perhaps using alternative algorithms and/or cryptographic schemes. Moreover, as another future project, we suggest assessing the impact of running the protocol in a concurrent setting, and perhaps even outsourcing the computations to third parties.

9. Bibliography

References

[1] Davide Andreoletti, Omran Ayoub, Silvia Giordano, Massimo Tornatore, and Giacomo Verticale. Privacy-preserving multi-operator contact tracing for early detection of covid19 contagions. pages 1–6. IEEE, 12 2020.

[2] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. 2016.

[3] Jung Hee Cheon, Yongha Son, and Donggeon Yhee. Practical fhe parameters against lattice attacks. *J. Korean Math. Soc*, 59:35–51, 2022.

[4] Craig Gentry. A fully homomorphic encryption scheme. 2009.

[5] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. 1978.

10. Acknowledgements

Massimo Tornatore, Davide Andreoletti, Omran Ayoub, Qiaolun Zhang and my parents.