Privately Connecting Mobility to Infectious Diseases via Applied Cryptography

Alessandro Bruni[3], Lukas Helminger[1,2], Daniel Kales[1], Christian Rechberger[1], and Roman Walch[1,2]

[1] Graz University of Technology, Graz, Austria
[2] Know-Center GmbH, Graz, Austria
[3] Katholieke Universiteit Leuven, Leuven, Belgium

Abstract. Human mobility is undisputedly one of the critical factors in infectious disease dynamics. Until a few years ago, researchers had to rely on static data to model human mobility, which was then combined with a transmission model of a particular disease resulting in an epidemiological model. Recent works have consistently been showing that substituting the static mobility data with mobile phone data leads to significantly more accurate models. While prior studies have exclusively relied on a mobile operator's subscribers' aggregated data, it may be preferable to contemplate aggregated mobility data of infected individuals only. Clearly, naively linking mobile phone data with infected individuals would massively intrude privacy. This research aims to develop a solution that reports the aggregated mobile phone location data of infected individuals while still maintaining compliance with privacy expectations. To achieve privacy, we use homomorphic encryption, zero-knowledge proof techniques, and differential privacy. Our protocol's open-source implementation can process eight million subscribers in one hour. Additionally, we provide a legal analysis of our solution with regards to the General Data Protection Regulation.

Keywords: FHE, privacy, Covid-19, mobile data, GDPR

1 Introduction

1.1 Human Mobility and Infectious Diseases

Human mobility is undisputedly one of the critical factors in infectious disease dynamics. On the one side, increased human mobility may account for more contacts between receptive and infected individuals. On the other side, human travel may introduce pathogens into new geographical regions. Both cases can be responsible for an increased prevalence and even an outbreak of the infectious disease [55]. In particular, human travel history has been shown to play a critical role in the propagation of infectious diseases, like influenza [24] or measles [31]. Therefore understanding the spatiotemporal dynamics of an epidemic is closely tied to understanding movement patterns of infected individuals.

Mobile Phone Data. Until a few years ago, researchers had to rely on static data – relative distance and population distribution – to model human mobility, which was then combined with a transmission model of a particular disease resulting in an epidemiological model. This model was then used to improve the understanding of the geographical spread of epidemics. Mobile phones and their location data have the unique potential to improve these epidemiological models further. Indeed, recent work [57] has been showing that substituting the static mobility data with mobile phone data leads to significantly more accurate models. Integrating such up-to-date mobility patterns allowed them to identify hotspots with a higher risk of contamination, enabling policymakers to apply focused measures.

While prior studies have exclusively relied on a mobile operator's subscribers' aggregated data, it may be preferable to contemplate aggregated mobility data of infected individuals only. Indeed, a cholera study [25] observed that although their model has high accuracy, it performs less well when the cumulative incidence is low. They speculated that demographic stochasticity could be one reason for the bad performance of their model. In other words, the infected individuals' mobility pattern may not be precisely reflected by the population's mobility if the prevalence is low. In order to mitigate this problem, we propose the usage of infected individuals' mobile phone data, which should lead to an improvement in the predictive capabilities of epidemiological models, especially in highly dynamic situations.

Privacy Issues. Clearly, naively linking mobile phone data with infected individuals would massively intrude privacy. Namely, either the mobile network operator would have to know which subscribers are infected, or, the epidemiological researchers would have to get access to non-anonymized data records. As a result, previous studies considered the availability of travel history information from patients as not possible and attempted to control possible biases in the results manually [51].

1.2 Our Contribution

This research aims to develop a software solution that reports the aggregated mobile phone location data of infected individuals while still maintaining compliance with privacy expectations. We use various state-of-the-art privacy-preserving cryptographic primitives to design a two-party protocol that achieves the following: The epidemiological researcher or a health authority inputs patients' identifiers, whereas the mobile operator inputs call detail records (CDRs) of its subscribers. The protocol outputs the patients' aggregated location data from the CDRs to the health authority. Informally, neither does the health authority access individuals' CDRs nor does the mobile operator learn which subscribers were involved in the computation, and therefore, who is infected.

To achieve the privacy goals outlined above, we use homomorphic encryption [26], zero-knowledge proof techniques [28], and differential privacy [21]. In particular, the patients' identifiers get homomorphically encrypted before sending them to the mobile operator. Due to the nature of homomorphic encryption, the mobile operator can perform the data aggregation without decrypting the identifiers. To prevent the health authority from learning individual CDRs, we ensure that the identifiers' set has a minimum cardinality by applying zero-knowledge proof techniques. In addition, the mobile operator can add noise - in the sense of differential privacy - to the aggregated CDRs before sending them to the health authority. This becomes necessary if the aggregated CDRs would still leak information that could lead to patients' re-identification. More formally, we defined our protocol as an ideal functionality, which is a common practice for secure computation protocols [12, 27]. We show input privacy in the presence of a maliciously controlled mobile operator provided that the homomorphic encryption scheme is semantically secure.

Our protocol's open source implementation is written in C++ using the SEAL [50] library and tested with parameters suitable for entire nation-states. In the beginning, we also explored whether classical multi-party computation [22] (secret sharing or garbled circuits) could be used for our protocol's realization, but the immense data complexity constituted a practical obstacle. Instead, we thoroughly optimized the homomorphic data aggregation phase. Now, our protocol can process eight million subscribers in one hour (corresponding to roughly 5 € using AWS).

In addition, we conducted a legal case study of our use case. More concretely, we focused on the EU General Data Protection Regulation (GDPR)[4] because the GDPR is one of the most strict privacy frameworks.

1.3 Road-map

The following sections not only contain a description of our solution but also a rigorous analysis regarding legal aspects, security and privacy. In Section 2, we discuss the relevant related work. Section 3 provides the necessary preliminary definitions and notations. Section 4 first states the problem we want to solve in this article. It then gradually develops a solution by introducing privacy protection mechanism step by step. Section 5 and Section 6 are dedicated to a thorough legal, security and privacy analysis of our solution. Section 7 elaborates on the implementation of our solution as well as demonstrating the performance. Section 8 concludes with a discussion about considerations for an actual roll-out. We defer to the appendix additional material on the GDPR (Appendix A), missing proofs of our security analysis (Appendix B), formal definitions for differential privacy (Appendix C), and details about the implementation (Appendix D).

[4] https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex:32016R0679

2 Related Work

Numerous research directions have previously sought to model the spread of infectious diseases. Most closely related to this paper is work connecting mobile phone data to infectious diseases.

Impact of Human Mobility on Infectious Diseases: The Use of Call Detail Records. Mobility data derived from call detail records (CDRs) – phone calls and text messages – have been used to understand various infectious diseases' spatial transmission better, see Table 1. There is a general understanding that – although not perfect – mobile phone data provide an opportunity to model human travel patterns and thereby enhance the understanding of the transmission of infectious diseases [55].

Each of the studies got their CDRs from one mobile operator. Most of the time, this mobile operator had the largest market share and coverage. The common understanding is that biases such as Multi-SIM activity and different mobile phone usage across different geographical and socio-economic groups have a limited effect on general estimates of human mobility [56].

	Disease	Country	Year of dataset	Subscribers (millions)	Period (months)
[52]	Malaria	Tanzania	2008	0.8	3
[57]	Malaria	Kenya	2008-09	14.8	12
[36]	HIV	Kenya	2008-09	14.8	12
[58]	Rubella	Kenya	2008-09	14.8	12
[4]	Cholera	Haiti	2010	2.9	2
[51]	Malaria	Namibia	2010-11	1.5	12
[59]	Dengue	Pakistan	2013	39.8	7
[25]	Cholera	Senegal	2013	0.1	12

Table 1: Studies connecting mobile phone data to infectious diseases.

The most common model was to assign an individual a daily location. More concretely, each subscriber was assigned to a study area on each day based on the cell tower with the most CDRs or the last outgoing CDR. Further, the primary study area ("home") was computed for each subscriber by taking the study area where the majority of days were spent. A slightly different approach was to assign each subscriber to a study area using the last outgoing CDR of each day, and not considering a primary study area [4]. A more refined approach was to compute the number of CDRs made for every subscriber in each study area [25]. The primary study area was defined to be the study area where with the most CDRs between 7 p.m. and 7 a.m.

All of the studies emphasized that preserving individuals' privacy is mandatory. Each of the studies applied anonymization and aggregation as privacy measures. More concretely, in all cases - to the best of our understanding - the involved mobile operator anonymized the CRDs before handing them over to the health authority. In addition, we found that the mobile network operator aggregated the CDRs in at least two cases. However, none of the studies discussed privacy' definitions or the potential risk of de-identification, which is especially high for location data [38]. Therefore, it is hard to assess if they achieved their goal of preserving individuals' privacy in the studies.

Automatic Contact Tracing. Due to the ongoing global threat of Covid-19, a number of technological approaches are currently developed to help reduce its spread and impact. A lot of focus is on automatic contact tracing, where the main challenges include privacy-friendliness, scalability and utility. Numerous efforts to improve privacy-friendly contact tracing exist, including [20, 6, 13, 5, 7, 14, 53, 1, 30, 46, 54], among others.

These approaches crucially rely on sizable parts of the population using smartphones, enabling Bluetooth, and installing a new App on their phones. In contrast, our proposal does not help with contact tracing, but gives potentially useful epidemiological information to health authority without requiring people to carry around smartphones, as any mobile phone will be sufficient. Furthermore, our solution does not require people to enable Bluetooth on their phones.

3 Preliminaries

In this section, we cover the preliminaries required for the rest of the paper. We will first introduce the notations we use in the rest of the paper, before we describe homomorphic encryption, and differential privacy.

3.1 Notation

We follow the widespread convention to write vectors in bold lower case letters and matrices in upper case letters. We use x_i to access the i-th element of vector $\boldsymbol{x}$. For $m \in \mathbb{N}$ and $x \in \mathbb{Z}$, let $\boldsymbol{x}^m$ be defined as the vector of powers of x: $\boldsymbol{x}^m = (1, x^1, ..., x^{m-1})$. We denote by $\boldsymbol{c} \circ \boldsymbol{d}$ the element-wise multiplication (Hadamard product) of the vectors $\boldsymbol{c}$ and $\boldsymbol{d}$. For a positive integer t, we identify $\mathbb{Z}_t = \mathbb{Z} \cap [-t/2, t/2)$. For a real number r, $\lfloor r \rceil$ denotes the nearest integer to r, rounding upwards in case of a tie.

3.2 Homomorphic Encryption

The concept of homomorphic encryption (HE) has often been considered to be the holy grail in cryptography since it allows us to work on encrypted data without requiring the secret decryption key. It was first introduced by Rivest et al. [48] and partially HE schemes, i.e. schemes which allow performing a limited set of operations on encrypted data, have been known for years: The RSA [49] encryption scheme is homomorphic for multiplication and Paillier's cryptosystem [45] is homomorphic for addition. However, it was not until Gentry's groundbreaking work from 2009 [26] that we were able to construct the first fully homomorphic encryption (FHE) scheme, a scheme which in theory can evaluate an arbitrary circuit on encrypted data. His construction is based on ideal lattices and is deemed to be too impractical ever to be used, but it led the way to construct more efficient schemes in many following publications [9, 8, 23, 16, 17].

Modern HE schemes are based on the learning with errors (LWE) [47] hardness assumption, and its variant over polynomial rings, the ring learning with error (RLWE) [42] hardness assumption. During the encryption of a plaintext in RLWE based schemes, random noise is introduced into the ciphertext. This noise grows with the evaluation of homomorphic operations, negligible for addition, but significantly for homomorphic multiplication. Once this noise becomes too large and exceeds a threshold, the ciphertext cannot be decrypted correctly anymore. We call such a scheme a somewhat homomorphic encryption scheme (SHE), a scheme that allows evaluating an arbitrary circuit over encrypted data up to a certain depth. The specific depth depends on the choice of encryption parameters, and choosing parameters for larger depths comes, in general, with a considerable performance penalty.

In his work [26], Gentry introduced the novel bootstrapping technique, a procedure that reduces the noise in a ciphertext and can turn a (bootstrappable) SHE scheme into an FHE scheme. However, this bootstrapping operation comes with high computational complexity. In many practical applications it is, therefore, faster to omit bootstrapping and choose a SHE scheme with large enough parameters to evaluate the desired circuit. In this work, we use the BFV [8, 23] SHE scheme to homomorphically encrypt the inputs of our protocol.

Homomorphic Encryption vs. Generic MPC. We rely on HE instead of other privacy-preserving protocols, such as secure multi-party computation (MPC), due to several considerations:

- Homomorphic ciphertext-ciphertext multiplications are very costly in HE schemes, however, in our protocol we mainly rely on the cheaper plaintext-ciphertext multiplications. Therefore, all the operations involved in our protocol can be expressed relatively cheap using HE.
- HE has the advantage of outsourcing computations. After the client sends the encrypted data to the server, the server can do the computations without further data exchange with the client. MPC protocols based on secret-sharing, in contrary, have a higher number of communication rounds and all parties have to participate in the computations.
- Generic MPC protocols are not well suited for the large databases considered in this work. Both, secret sharing and garbled circuit based MPC, would require the (secure) transmission of the server's database (either in secret-shared form or embedded in a circuit) to the client, requiring several GB of communication (e.g., $2^{23} \times 2^{15}$ matrix of 32 bit integers has a size of 1024 GB). Furthermore, in the most efficient secret sharing schemes, such as the popular SPDZ [18, 19], the multiplication of two shared values requires a shared

beaver-triple which has to be precomputed in an expensive offline phase and can not be reused for further computations. However, computing enough triples to support the secure aggregation in our protocol, i.e., one triple per database entry, would require extensive runtime and communication. For example, on our benchmarking platform, generating 2^{20} triples (enabling the same number of secure multiplications) using the MP-SPDZ [37] library in a semi-honest security setting already took 100 seconds in a LAN-setting and required 4 GB of communication.

3.3 Differential Privacy

When we design privacy-preserving data analytics protocols, we have to consider that the result, can still leak too much information about the underlying dataset. In our case – a protocol designed to aggregate location data – the result could still leak the location of a single individual [60]. We can use the well-established notion of differential privacy [21] to help protect against such kind of information leakage.

Differential privacy defines a robust, quantitative notion of privacy for individuals. The main idea is that the outcome of a computation should be as independent as possible from the data of a single individual. This independence can be parameterized, usually denoted by the privacy parameter ϵ.

We opted for differential privacy for its compatibility with existing privacy frameworks as well as the success in several real-world applications. Recent work [44] showed that differential privacy satisfies privacy requirements set forth by FERPA[5]. Even before this analysis, several businesses were already using differential privacy. For example, Apple [3] and Google [29] have applied differential privacy to gather statics about their users without intruding on individual users' privacy. The U.S. Census Bureau announced that the 2020 Census will use differential privacy as a privacy protection system [11]. These examples highlight that despite being a relatively new concept, differential privacy is already well-established.

The most prevalent technique to achieve differential privacy is to add noise to the outcome of the computation. In this article, we construct the noise from a zero-centered Laplace distribution. The distribution is calibrated with a privacy parameter ϵ and the global sensitivity Δq of the computation and has the following probability density function:

$$Lap(x|b) = \frac{1}{2b} e^{-\frac{|x|}{b}}, \qquad \text{with } b = \frac{\Delta q}{\epsilon}$$

To add differential privacy to a protocol operating on integers, we discretize the Laplace distribution by rounding the sampled value to the nearest integer. For a formal definition of differential privacy, we refer to Appendix C.

[5] Family Educational Rights and Privacy Act of 1974, U.S.

4 Problem Statement and Solution

In this section, we first discuss our protocol in plain without measures to protect involved data, before we introduce each privacy protection mechanism step by step. We provide a formal security analysis of the final protocol in Section 6.

4.1 The Plain Protocol

In this work, we want to accumulate the location data of infected individuals to create a heatmap of places with higher risk of getting infected, assisting governments in controlling an epidemic. For this purpose, two parties controlling two different datasets are involved: Health authority who knows which individuals are infected; and a mobile operator who knows location data of their subscribers. More specifically, the mobile operator knows how long each of their subscribers is connected to which cell towers, and therefore, an approximated location data. The final heatmap will then show, how much time infected individuals spent in which area, and therefore, will show areas with higher chance of getting infected with the disease.

If the mobile operator knows which of its subscribers is infected, it can do the following to create the desired heatmap:

- Initialize a vector $\boldsymbol{h}$ of k elements with zeros, where k is the total number of cell towers. Each element of this vector corresponds to one cell tower.
- For each infected individual, add the amount of time it spent at each cell tower to the corresponding element of the vector $\boldsymbol{h}$.
- After all individuals are processed, the vector $\boldsymbol{h}$ contains the final heatmap, i.e. h_j contains the accumulated time spent of all infected individuals at cell tower j.

Now let us rewrite this process into a single matrix multiplication. First we encode all N subscribed individuals into a vector $\boldsymbol{x} \in \mathbb{Z}_2^N$, with $x_i \in \mathbb{Z}_2$ indicating, whether the individual i is infected ($x_i = 1$) or not ($x_i = 0$). Then we encode the location data in a matrix $Z = (\boldsymbol{z_1}, \boldsymbol{z_2}, \ldots, \boldsymbol{z_k}) \in \mathbb{Z}^{N \times k}$ such that the vector $\boldsymbol{z_j}$ contains all the location data corresponding to the cell tower identified by j. In other words, the i-th element of the vector $\boldsymbol{z_j}$ contains the amount of time the individual i spent at cell tower j. Now we can calculate the heatmap as $\boldsymbol{h} = \boldsymbol{x}^T \cdot Z$.

We depict the basic protocol, involving the health authority as a client and the mobile operator as a server, in Figure 1, assuming the health authority and the mobile phone operators already agree on identifying all subscribed individuals by indices $i \in 1, ..., N$.

Remark 1 (Agreeing on database indices). The protocol in Figure 1 already assumes that the two parties agree on the indices of individuals in the database. In practice, the individuals would likely be identified by their phone numbers. We now give two options to get a mapping from a phone number to a database index:

- The server sends a mapping of all phone number to their database index in plain. This approach is simple and efficient, but it discloses the list of all subscribed individuals to the client. However, this list is essentially a list of all valid phone numbers in random order and does not leak anything more than the validity of that number. Still, this may be an issue in some scenarios.
- The server and client engage in a protocol for Private Set Intersection with associated data (e.g., [15]). In such a protocol the client and the server input their list of phone numbers and the client gets as the output of the protocol the phone numbers that are in both sets, as well as associated data from the server side, which in our case would be the index in the database.

While the PSI-based solution has some overhead compared to the plain one, the performance evaluation in [15] shows that a protocol execution with 2^{24} server items and 5535 client items takes about 22 seconds with a total communication of 17 MB – a minor increase when looking at the overall protocol.

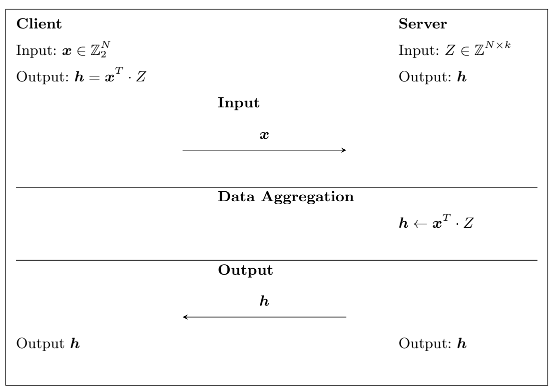

Fig. 1: Basic Protocol without privacy protection.

Simply executing the protocol described in Figure 1 of course would enable the mobile operator to learn which individuals are infected, which is a huge privacy violation. On the other hand, the health authority could query a single individual's location data by sending a vector $\boldsymbol{x} = (1, 0, \dots, 0)$, which also violates privacy. Furthermore, a correctly accumulated heatmap still can leak some information about individuals location data. In the following, we describe our techniques to protect against these privacy violations.

4.2 Adding Encryption

To protect the vector send by the health authority, and therefore who is infected and who is not, we use a homomorphic encryption (HE) scheme $\mathsf{HE} =$

$(\mathsf{HE.KGen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$. Before executing the protocol, the health authority runs KGen to obtain a secret key sk and an evaluation key evk. We assume that the mobile operator knows evk, which is required to perform operations on encrypted data, before running the protocol.

In the updated protocol, the health authority now uses sk to encrypt the input vector $\boldsymbol{x}$ and sends the resulting ciphertext vector $\boldsymbol{c} \leftarrow \mathsf{HE.Enc}_{\mathsf{sk}}(\boldsymbol{x})$ to the mobile operator. The mobile operator then uses evk to perform the matrix multiplication on the encrypted input vector and sends the resulting ciphertext vector $\boldsymbol{h}^* \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\boldsymbol{c}^T \cdot Z\right)$ back to health authority. The health authority can now use sk to decrypt the result and get the final heatmap $\boldsymbol{h} = \mathsf{HE.Dec}_{\mathsf{sk}}(\boldsymbol{h}^*)$.

Informally, if the used HE scheme is semantically secure, then the mobile operator cannot learn which individuals are infected by the disease and which are not.

4.3 Invalidation Results for Malicious Queries

In the simple protocol, the health authority could use the input vector $\boldsymbol{x}$ to get information about the location data of individuals. Since the input vector is encrypted, the mobile operator cannot trivially check if the vector is malicious or not. Also, comparing encrypted elements is not trivially possible in most HE schemes. However, we can encode all the required checks to output 0, if everything is correct, and a random value otherwise. We then can add this value to the final output as a masking value which randomizes the mobile operator response if the input vector is malicious. We describe how to generate this masking value for different proofs in the following sections.

Masking for Non-Binary Query Vector. The aim of this mask is to ensure that an infected individual's location data gets aggregated not more than once. Note that, the HE scheme we use operates on plaintexts in $\mathbb{Z}_t$, i.e., integers modulo t. Therefore, the inputs to our protocol, i.e., the vector $\boldsymbol{x}$ and the matrix Z consist of elements in $\mathbb{Z}_t$. As lined above it is crucial to the protocol's privacy that input vector is binary, i.e., only contain 0s and 1s. If this is not the case, the client can arbitrarily modify the contribution of a single person to the overall aggregated result, which can leak private information. Since the server only receives an encryption of the input vector, simply checking for binary values is not an option.

However, we can use similar techniques to the ones used in Bulletproofs [10] to provide assurance that the query vector $\boldsymbol{x} \in \mathbb{Z}_t^N$ only contains binary elements. First, we will exploit the following general observation. Let $\boldsymbol{d} = \boldsymbol{x} - \mathbf{1}$, then $\boldsymbol{x} \circ \boldsymbol{d}$ is the zero vector if $\boldsymbol{x}$ is binary. Note in our scenario, $\boldsymbol{d}$ can be computed by the server. The result of Hadamard product $\boldsymbol{x} \circ \boldsymbol{d}$ can be aggregated into a single value by calculating the inner product $\langle \boldsymbol{x}, \boldsymbol{d} \rangle$, which will again be zero if $\boldsymbol{x}$ is binary. The server then adds a random value y to reduce the probability for the client to cheat by letting several entries of $\boldsymbol{x}$ cancel each other out during the inner product, which gives the mask:

$$\mu_{\mathtt{bin'}} = \langle \boldsymbol{x}, (\boldsymbol{d} \circ \boldsymbol{y}^N) \rangle \,. \tag{1}$$

For the generic case of a vector $\boldsymbol{x}$ and a randomly chosen y, $\langle \boldsymbol{x}, \boldsymbol{y}^N \rangle = 0$ will hold for a $\boldsymbol{x} \neq 0$ only with probability N/t [10]. Using a ν bit t ($t \approx 2^\nu$), translates to a soundness error of $\nu - \log_2(N)$ bits, for details of this calculation see Appendix B.1. In particular, if we look at $N = 2^{23}, \nu = 60$, parameters sufficient for small nation-states (see Section 7.6), we get 37-bit statistical security. Standard literature suggest a statistical security parameter of at least 40-bit; therefore, we developed a method to enhance the statistical security without significant overhead.

Boosting Soundness for Non-Binary Query Vector Mask. The high level idea is that we lower the probability of cheating successfully by independently checking the above mask twice. We extended the previous mask to the following:

$$\mu_{\mathtt{bin}} = \langle \boldsymbol{x}, (\boldsymbol{d} \circ \boldsymbol{y_1}^N) \rangle \cdot r_1 + \langle \boldsymbol{x}, (\boldsymbol{d} \circ \boldsymbol{y_2}^N) \rangle \cdot r_2$$

where $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_t \setminus \{0\}$ are two random values. Therefore, the statistical security level increases to $\nu - 1$-bit ($= 59\,\text{bit}$). We refer to Lemma 2 for a proof of this statement.

Masking Against Wrong Hamming Weight. Another problem of our protocol is that the client can target the values of single individuals by querying the server with an input vector of hamming weight one. Again, since the query is encrypted, the server can not trivially check the hamming weight of the input vector. However, we can apply similar techniques as in the previous section to incorporate a hamming weight check into a masking value.

Again let $\boldsymbol{x}$ be the query vector and let w be its announced hamming weight. On the server-side, calculate $\langle \boldsymbol{x}, \boldsymbol{1}^N \rangle$, which is equal to the hamming weight of $\boldsymbol{x}$. Therefore, the following mask $\mu_{\mathtt{HW}} \in \mathbb{Z}_t$ is zero if $\boldsymbol{x}$ has the announced hamming weight w:

$$\mu_{\mathtt{HW}} = \langle \boldsymbol{x}, \boldsymbol{1}^N \rangle - w \,.$$

We note, that $\mu_{\mathtt{HW}}$ is controlled and known by the client.

Applying the Masks. Once the final mask is calculated, it gets added to the final output of the protocol. However, in case the masking value is not zero, we have to make sure that a different random value is added to each element of the output vector to prevent leaking the mask if some values of the output vector are known beforehand. Therefore, the final mask $\boldsymbol{\mu}$ can be calculated using a random vector $\boldsymbol{r} \xleftarrow{\$} (\mathbb{Z}_t \setminus \{0\})^k$ as follows:

$$\boldsymbol{\mu} = (\mu_{\mathtt{bin}} + \mu_{\mathtt{HW}}) \cdot \boldsymbol{r} \tag{2}$$

$\boldsymbol{\mu}$ is now equal to $\mathbf{0}^k$ if $\boldsymbol{x}$ is a binary vector with hamming weight w, random otherwise. The whole procedure reduces the statistical security of our protocol by one bit. Hence, our protocol enjoys $\nu-2$ bit (= 58-bit) security, see Appendix B.1 for a proof.

Remark 2. Adding the hamming weight check into the proving mask inherently leaks the number of infected individuals in the query. Since the number of infected individuals is usually public, this does not represent a problem. Nevertheless, one could omit the hamming weight check by setting $\boldsymbol{\mu} = \mu_{\mathtt{bin}} \cdot \boldsymbol{r}$, getting a protocol with $\nu - 1$ bit statistical security.

4.4 Adding Differential Privacy

Even with a cardinality check in place, the final heatmap can still leak information about location data of individuals. As an example, the health authority could abuse the heatmap to track an individual by querying him alongside individuals from a completely different area. The location data of the targeted individual would be clearly visible as an isolated zone in the resulting heatmap. Applying differential privacy with suitable parameters will protect against such an attack since the overall goal of differential privacy is to decrease the statistical dependence of the final result to a single database entry. In our use-case, therefore, differential privacy achieves that it is infeasible to distinguish between heatmaps, in which we include a single individual in the accumulation and heatmaps in which we do not.

Choosing proper parameters, however, highly depends on the underlying dataset. On the one hand, the chosen ϵ should be small enough to satisfy privacy concerns; on the other hand, it should be big enough not to overflow the result with noise. In our protocol, accumulations of a sufficient amount of individuals should not be affected by the noise, i.e., the noise on its own should not be able to create hotspots.

4.5 Final Protocol

Finally, with all measures to protect privacy in place, we present the final protocol in Figure 2.

5 Legal Case Study

For the social context and background considerations regarding this legal case study we refer to Appendix A.

In our use case, data held by both the Health Authority[6] and the Electronic Communication Service providers fall both in the definition of personal data. On the one hand, data in possession of the Health Authority, namely personal

[6] The Health National Authority is the entity empowered to carry out and enforce policy measures approved at EU and national touching upon the health sector.

Client		Server
Input: $\boldsymbol{x} \in \mathbb{Z}_t^N$		Input: $Z \in \mathbb{Z}_t^{N \times k}$
Output: If $\boldsymbol{x} \in \mathbb{Z}_2^N : \boldsymbol{h} = \boldsymbol{x}^T \cdot Z + \left(\left\lfloor Lap\left(\frac{\Delta q}{\epsilon}\right)\right\rceil\right)^k \in \mathbb{Z}_t^k$		Output: $\perp$
Otherwise: $\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_t^k$		
	Encryption	
$\boldsymbol{c} \leftarrow \mathsf{HE.Enc}_{\mathsf{sk}}(\boldsymbol{x})$		
$w \leftarrow \langle \boldsymbol{x}, 1^N \rangle$		
	$\xrightarrow{\boldsymbol{c}, w}$	
	Data Aggregation	
		$\boldsymbol{h}^* \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\boldsymbol{c}^T \cdot Z\right)$
	Compute Mask	
		$\boldsymbol{d} \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\boldsymbol{c} - \mathbf{1}^N\right)$
		$\boldsymbol{r} \xleftarrow{\$} \mathbb{Z}_t^k; r_1, r_2, y_1, y_2 \xleftarrow{\$} \mathbb{Z}_t$
		$\mu_{\mathtt{bin}} \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\langle \boldsymbol{c}, (\boldsymbol{d} \circ \boldsymbol{y_1}^N)\rangle r_1 + \langle \boldsymbol{c}, (\boldsymbol{d} \circ \boldsymbol{y_2}^N)\rangle r_2\right)$
		$\mu_{\mathtt{HW}} \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\langle \boldsymbol{c}, \mathbf{1}^N \rangle - w)\rangle\right)$
		$\boldsymbol{\mu} \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left((\mu_{\mathtt{bin}} + \mu_{\mathtt{HW}}) \cdot \boldsymbol{r}\right)$
		$\boldsymbol{h}^* \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}\left(\boldsymbol{h}^* + \boldsymbol{\mu}\right)$
	Differential Privacy	
		$\boldsymbol{\delta} \xleftarrow{\$} \left(\left\lfloor Lap\left(\frac{\Delta q}{\epsilon}\right)\right\rceil\right)^k$
		$\boldsymbol{h}^* = \mathsf{HE.Eval}_{\mathsf{evk}}\left(\boldsymbol{h}^* + \boldsymbol{\delta}\right)$
	Decryption	
	$\xleftarrow{\boldsymbol{h}^*}$	
$\boldsymbol{h} = \mathsf{HE.Dec}_{\mathsf{sk}}(\boldsymbol{h}^*)$		
Output $\boldsymbol{h}$		

Fig. 2: Final protocol.

medical data and on the other hand, also call details, data records and call detail record (CDR) fall in the definition provided by the GDPR. As a result, any processing activity carried out by these two entities on such data should be considered as falling in the scope of application of the GDPR. Thus, such processing activities have to fall the requirements listed in GDPR and comply with privacy and data protection principles, listed in Art. 5 GDPR.

5.1 Roles

In the EU Privacy and Data Protection subjects involve in activities that fall into the definition of processing[7] of personal data are three: controller, processor[8] and data subjects. In a recent guideline on the role of controller and processor the European Data Protection Board (EDPB),[9] in line with the previous Article 29 Working Party (WP29) Opinion[10] has pointed out the main characteristics of controller role. According to the EDPB one of the crucial element necessary to identify a controller concern its *'factual influence that the controller has over the processing operation, by virtue of an exercise of decision-making power.'*[11] In our use-case, domestic legislation might have delegated specific activities to the health authority in order to develop a comprehensive strategy to fight the COVID-19 crisis.

The WP29 Opinion had already clarified that in case a controller(travel agency) would have shared personal data to other entities (hotels), the entity in possession of these personal data should have to be configured together with the travel agency a controller, creating a joint-controllership[12] with him.[13]

From a legal perspective, to determine the nature and roles concerning the processing of personal data, an assessment of the activities is necessary.

5.2 Activities and Context

In the context of the given use case should be assessed whether the cryptographic techniques that have been used to encrypt different personal data sets have

[7] Art. 4(2) GDPR: 'processing' means any operation or set of operations which is performed on personal data or on sets of personal data, whether or not by automated means, such as collection, recording, organisation, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure or destruction;'

[8] Art.4(8) GDPR: 'processor' means a natural or legal person, public authority, agency or other body which processes personal data on behalf of the controller;'

[9] European Data Protection Board, Guidelines 07/2020 on the concepts of controller and processor in the GDPR, adopted on 2 September 2020, https://edps.europa.eu/sites/edp/files/publication/19-11-07_edps_guidelines_on_controller_processor_and_jc_reg_2018_1725_en.pdf

[10] Article 29 Working Party, Opinion 1/2010 on the concepts of "controller" and "processor" , adopted on 16 February 2010, WP169, http://ec.europa.eu/justice/policies/privacy/docs/wpdocs/2010/wp169_en.pdf.

[11] European Data Protection Board, Guidelines 07/2020 on the concepts of controller and processor in the GDPR, adopted on 2 September 2020, https://edps.europa.eu/sites/edp/files/publication/19-11-07_edps_guidelines_on_controller_processor_and_jc_reg_2018_1725_en.pdf, p.7

[12] Art. 26 GDPR

[13] Article 29 Working Party, Opinion 1/2010 on the concepts of "controller" and "processor" , adopted on 16 February 2010, WP169, http://ec.europa.eu/justice/policies/privacy/docs/wpdocs/2010/wp169_en.pdf, p.19

made the identification of data subjects no longer possible. If this is the case, the anonymised data fall out of the GDPR's scope of application, and involved actors will not have to comply with such rules.

According to WP29[13], and mentioned CJEU jurisprudence,[14] to assess the identifiability should be considered objective aspects such as time and technical means, together with other contextual elements. In such a context monitoring of latest developments in the anonymisation and re-identification attacks scenario results to be crucial, especially when the processing activity involves location data, known for being difficult to be anonymised.

In the given use case, the two main entities involved in the processing of personal data have used different cryptographic methods to make identifiers anonymous. Specifically, the Health authority has used homomorphic encryption for the names of COVID positive individuals while the electronic communication service provider has used differential privacy. The process to encrypt and make inaccessible, such data is considered by GDPR and Article 29 as a processing activity. Therefore, both entities applying such privacy-preserving technology should be considered as the data controller, and their processing activity should comply with GDPR and ePrivacy requirements. As a result, the encryption of data to make them anonymous should be carried out complying, among the others, the purpose limitation principle and should have a lawful basis if there is no compatibility between the first processing activity, namely, collection of data, and the anonymisation one.[15] Taking into account our research and the objective of the activities carried out in such context, we should mention Art. 23 GDPR. The GDPR, which is flexible that foreseen situations where the fundamental right of data protection might be limited, offer an exception to data protection principle. This situation should be possible only when Union or Member state law foreseen a restriction to data controller obligation *'when such a restriction respects the essence of the fundamental rights and freedoms and is a necessary and proportionate measure in a democratic society to safeguard'*.[16] Art. 23(5) GDPR also mentions public health and social security as one of the grounds that can justify a restriction of privacy and data protection obligations. Therefore, the processing activity of both electronic communication operator and health authority to anonymised, through privacy-enhancing-technologies (so-called PETs), personal data should consider in compliance with GDPR provisions.[16]

After having assessed the compliance of encryption activity with the EU privacy and data protection framework, additional consideration should be done, namely, if the encryption methods used by the two entities can be defined anonymous and consequently which are the obligations for the Health Authority and

[14] Patrick Breyer v Bundesrepublik Deutschland [2016] European Court of Justice Case C-582/14, ECLI:EU:C:2016:779 [46]

[15] Gerald Spindler and Philipp Schmechel, 'Personal Data and Encryption in the European General Data Protection Regulation' [2016] Journal of Intellectual Property, Information Technology and Electronic Commerce Law 15, 166; Finck and Pallas (n 23) 17–18.

[16] Art. 23 GDPR

the electronic communication operator rising from the EU privacy and data protection framework.

In conclusion, the analysis on the encryption methods used both by the health authority has to assess whether or not the two protocols used to allow any other entity to likely identify personal data from subjects other than the data controller.

6 Security Analysis

In this section, we show that our protocol is secure against semi-honest adversaries while providing privacy against a malicious server.

Two-party protocols are usually proven secure with the real-ideal world paradigm. Roughly speaking, one has to prove that the protocol does not leak any additional information than when computed with the help of a trusted third party. The trusted third party is modeled as an ideal functionality presented in Figure 3.

$\mathcal{F}_{CoV}$

Parameters: $t, N, W \in \mathbb{N}, \beta \in \mathbb{R}_+$. Running with parties P_1 and P_2.

1. Upon receiving an input $(\mathsf{input}, sid, P_1, P_2, x)$ from a party P_1, verify that $\boldsymbol{x} \in \mathbb{Z}_t^N$, else ignore input. Next, record $(sid, P_1, P_2, \boldsymbol{x})$. Once $\boldsymbol{x}$ is recorded, ignore any subsequent inputs of the form $(\mathsf{input}, sid, P_1, P_2, \cdot)$ from P_1.
2. Upon receiving an input $(\mathsf{input}, sid, P_1, P_2, Z)$ from party P_2, verify that $Z \in \mathbb{Z}_t^{N \times *}$, else ignore input. Proceed as follows: If there is a recorded value $(sid, P_1, P_2, \boldsymbol{x})$, compute $\boldsymbol{h} \leftarrow \boldsymbol{x}^T \cdot Z + (\lfloor Lap\,(\beta) \rceil)^k$ provided that $\boldsymbol{x} \in \mathbb{Z}_2^N$ and hamming weight of $\boldsymbol{x}$ is W, otherwise $\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_t^k$, and send (sid, P_1, P_2, k) where k is the number of columns of Z to the adversary. Then output $(\mathsf{result}, sid, P_1, P_2, \boldsymbol{h})$ to P_1. and ignore subsequent inputs of the form $(\mathsf{input}, sid, P_1, P_2, \cdot)$ from P_2.

Fig. 3: Ideal functionality $\mathcal{F}_{CoV}$ of the above solution.

First, we will show that our protocol is secure in the presence of semi-honest adversaries.

Lemma 1. *Let us assume* HE *is an IND-CPA secure homomorphic encryption scheme. Then protocol Figure 2 securely realizes* $\mathcal{F}_{CoV}$ *against static semi-honest adversaries.*

The high-level idea is that we reduce our protocol's security to the semantic security of the underlying homomorphic encryption scheme. Since by the definition of semantic security the server can not learn anything from encrypted data. The formal proof builds upon secure function evaluation and can be found in Appendix B.

Achieving simulation based security against a malicious server would be similar to verified homomorphic encryption. While some theoretical constructions exist [39], they are far from practical.

Instead, we show input privacy against a malicious server, which is also known as one-sided simulation security. This notion has been first considered in the context of oblivious transfer [43], was then formalized [35] and recently used [15] in the realm of PSI. Applied to the use-case at hand one-sided simulation guarantees that the patient's identifier are protected even in the presence of a malicious server (one that deviates from the protocol). For a formal definition see Appendix B.

Theorem 1. *Let us assume* HE *is an IND-CPA secure homomorphic encryption scheme. Then protocol Figure 2 securely realizes* $\mathcal{F}_{CoV}$ *with one-sided simulation in the presence of a maliciously controlled server.*

Proof. From Lemma 1, we already know that the protocols is secure against semi-honest adversaries. The only thing left to show is the input privacy of the client against a malicious server, i.e., the server is not able to learn any information from the client's input (patients' identifier). Now, due to the fact that server's view only includes a homomorphic encryption of the client's input, by the semantic security of HE we have that the server learns nothing about client's input. □

7 Implementation and Performance

We implemented our protocol using the BFV [8, 23] homomorphic encryption scheme, more specifically its implementation in the SEAL v3.4 [50] library. SEAL is an actively developed open-source library maintained by Microsoft Research compatible with all major operating systems, including Windows, Linux, and OS X.

The computationally most expensive phase in the protocol is the Data Aggregation phase, in which the server multiplies a huge matrix to a homomorphically encrypted input vector. Therefore, the main objective of our implementation is to perform this huge matrix multiplication as efficiently as possible.

7.1 Packing

Modern HE schemes allow for packing a vector of n plaintexts into only one ciphertext. Performing an operation on this ciphertext then is implicitly applied to each slot of the encrypted vector, similar to single-instruction-multiple-data (SIMD) instructions on modern CPU's (AVX, SSE2, etc.). However, the size of the ciphertext does not depend on the exact number ($\leq n$) of plaintexts encoded. The HE schemes support a variety of SIMD operations, including slot-wise addition, subtraction and multiplication, and slot-rotation. However, one can not directly access a specific slot of the encoded vector. We can use the SIMD encoding to speed up the matrix multiplication of our protocol significantly.

In the BFV scheme (and its implementation in the SEAL library), packing requires, that plaintexts are in $\mathbb{Z}_p$ with a prime p which is congruent to 1 mod $2 \cdot n$. The number of available SIMD slots is then equal to the degree of the cyclotomic reduction polynomial $(x^n + 1)$; thus, it is always a power of two. In the ciphertexts, the n slots are arranged as matrix of dimensions $(2 \times n/2)$. A ciphertext rotation affects either all rows, or all columns of the matrix simultaneously. Therefore, we can think of the inner matrix as two rotatable vectors, which can be swapped.

7.2 Baby-Step Giant-Step Matrix Multiplication

The SIMD encoding can be used to efficiently speed up matrix multiplication by using the diagonal method introduced by Halevi and Shoup in [32]. They have shown that a matrix-vector multiplication of a matrix $Z \in \mathbb{Z}^{m \times m}$ and vector $\boldsymbol{x} \in \mathbb{Z}^m$ can be expressed by m elementwise vector-vector multiplications, $m-1$ rotations, and $m-1$ additions, operations that can easily be evaluated in an HE scheme:

$$Z \cdot \boldsymbol{x} = \sum_{i=0}^{m-1} \mathtt{diag}(Z, i) \circ \mathtt{rot}(\boldsymbol{x}, i) \tag{3}$$

$\mathtt{diag}(Z, i)$ in equation 3 expresses the i-th diagonal of matrix Z in a vector of size m and $\mathtt{rot}(\boldsymbol{x}, i)$ rotates the vector $\boldsymbol{x}$ by index i to the left.

However, rotations are very expensive in terms of computational effort in the BFV encryption scheme. Luckily, the diagonal method can further be improved by applying the baby-step giant-step algorithm [33, 34]:

$$\begin{aligned} Z \cdot \boldsymbol{x} &= \sum_{i=0}^{m-1} \mathtt{diag}(Z, i) \circ \mathtt{rot}(\boldsymbol{x}, i) \\ &= \sum_{k=0}^{m_2-1} \sum_{j=0}^{m_1-1} \mathtt{diag}(Z, km_1 + j) \circ \mathtt{rot}(\boldsymbol{x}, km_1 + j) \\ &= \sum_{k=0}^{m_2-1} \mathtt{rot}\left(\sum_{j=0}^{m_1-1} \mathtt{diag}'(Z, km_1 + j) \circ \mathtt{rot}(\boldsymbol{x}, j), km_1 \right) \end{aligned} \tag{4}$$

where $m = m_1 \cdot m_2$ and $\mathtt{diag}'(Z, i) = \mathtt{rot}\left(\mathtt{diag}(Z, i), -\lfloor i/m_1 \rfloor \cdot m_1\right)$.[17] Note, that $\mathtt{rot}(\boldsymbol{x}, j)$ only has to be computed once for each $j < m_1$, therefore, equation 4 only requires $m_1 + m_2 - 2$ rotations of the vector $\boldsymbol{x}$ in total.

Trivially, we can use the following equation to implement a $\boldsymbol{x}^T \cdot Z$ multiplication, like we use in our protocol:

$$\begin{aligned} (\boldsymbol{x}^T \cdot Z)^T &= Z^T \cdot \boldsymbol{x} \\ &= \sum_{k=0}^{m_2-1} \mathtt{rot}\left(\sum_{j=0}^{m_1-1} \mathtt{diag}'(Z^T, km_1 + j) \circ \mathtt{rot}(\boldsymbol{x}, j), km_1 \right) \end{aligned} \tag{5}$$

[17] In equation 4, $\lfloor i/m_1 \rfloor$ is equal to k.

7.3 Homomorphic $N \times k$ Matrix Multiplication

In our protocol we want to homomorphically evaluate $\boldsymbol{x}^T \cdot Z$, where $\boldsymbol{x} \in \{0,1\}^N$ and $Z \in \mathbb{Z}_p^{N \times k}$, for big parameters N and k. As described in Section 7.1, the inner structure of the BFV ciphertext consists of two vectors of size $n/2$ each, and it does not allow a cyclic rotation over the whole input vector of size n. However, a rotation over the whole input vector is required by the baby-step giant-step algorithm. Therefore, we only can perform a baby-step giant-step multiplication with a $(n/2 \times n/2)$ matrix using this packing. Fortunately, we can use the remaining $n/2$ slots (i.e., the second vector in the inner structure of the BFV ciphertext) to perform a second $(n/2 \times n/2)$ matrix multiplication simultaneously. Therefore, after a homomorphic baby-step giant-step matrix multiplication, the result is a ciphertext c, where each of the two inner vectors encodes the result of a $(1 \times n/2) \times (n/2 \times n/2)$ vector-matrix multiplication. The sum of those two vectors can easily be obtained by rotating the columns of the ciphertext c and adding it to the first result:

$$c_{sum} = c + \texttt{rot}_{\texttt{col}}(c) \tag{6}$$

Thus, we can use one $(n/2 \times n/2)$ baby-step giant-step matrix multiplication and equation 6 to implement a homomorphic $(1 \times n) \times (n \times n/2) = (1 \times n/2)$ vector-matrix multiplication.

Taking this into account, we split the huge $(N \times k)$ matrix into $n_v \cdot n_o$ submatrices of size $(n \times n/2)$, with $n_v = \lceil \frac{N}{n} \rceil$ and $n_o = \lceil \frac{2k}{n} \rceil$, padding the submatrices with zeros if necessary. We split the input vector $\boldsymbol{x}$ into n_v vectors of size n (padding the last vector with zeros if necessary) and encrypt each of these vectors to get n_v ciphertexts c_i. The final result of the $\boldsymbol{x}^T \cdot Z$ matrix multiplication can be computed with the following equation:

$$\tilde{c}_i = \sum_{j=0}^{n_v - 1} \texttt{MatMul}(\texttt{SubMat}(Z, j, i)^T, c_j) \;\; \forall 0 \leq i < n_o \tag{7}$$

where, $\texttt{SubMat}(Z, j, i)$ returns the submatrix of Z with size $(n \times n/2)$, starting at row $n \cdot j$ and column $\frac{n}{2} \cdot i$, and $\texttt{MatMul}(Z, c)$ performs the homomorphic baby-step giant-step matrix multiplication $Z \cdot c$ followed by equation 6.

Equation 7 produces n_o ciphertexts $\tilde{c}_i$, with the final results being located in the first $n/2$ slots of the ciphertexts. Overall, our algorithm to homomorphically calculate $\boldsymbol{x}^T \cdot Z$ requires $n_v \cdot n_o$ baby-step giant-step matrix multiplications and the total multiplicative depth is 1 plaintext-ciphertext multiplication.

7.4 Homomorphic Evaluation of the Masking Value

To calculate the binary vector masking value (equation 1), we need to calculate the inner product of two homomorphically encrypted ciphertexts c and d. After an initial multiplication $c \cdot d$, the inner product requires $\log_2(n/2)$ rotations and addition, followed by equation 6 to produce a ciphertext, where the result is encoded in each of the n slots.

Our implementation uses rejection sampling and the SHAKE128 algorithm to cryptographically secure sample all the required random values in $\mathbb{Z}_p$. The total multiplicative depth to homomorphically evaluate the final mask (equation 2) is 1 ciphertext-ciphertext multiplication and 2 plaintext-ciphertext multiplications.

7.5 BFV Parameters

In BFV, one can choose three different parameters which greatly impact the runtime, security, and the available noise budget (i.e. how much further noise can be introduced until decryption will fail):

- Plaintext modulus t
  - In general an arbitrary integer t.
  - Needs to be prime and congruent to 1 (mod $2 \cdot n$) to enable packing.
- Ciphertext modulus $q = \prod_i q_i$, with q_i being prime.
- Degree n of the reduction polynomial (power of two).

We test our implementation for a computational security level of $\kappa = 128$ bit and $\kappa = 80$ bit. We use the LWE estimator [2] by Albrecht et al. to find suitable BFV parameters which provide 80 bit security against known attacks; for 128 bit security SEAL already provides parameters for different reduction polynomial degrees n. See Appendix D for more details on the impact of the parameters and which ones we used in our implementation.

7.6 Benchmarks

Multithreading. Since in our use cases N is much bigger than k, we implemented multithreading, such that the threads split the number of rows in the matrix (more specifically, the number of submatrices in the rows n_v) equally amongst all available threads. Therefore, each thread has to perform at most $\left\lceil \frac{n_v}{\#\text{threads}} \right\rceil \cdot n_o$ `MatMul` evaluations, which will be combined at the end by summing up the intermediate results. In case we want to add the mask to the result, an extra thread will perform the mask-evaluation in parallel to the matrix multiplication.

Benchmark Platform. Our prototype implementation[18] is compatible to Linux and Windows; however, we ran our benchmarks on a Linux cluster with two Intel Xeon E5-2699 v4 CPU's (total of 44 cores @ 2.2 GHz, 88 threads) and 512 GB RAM available.

[18] The source code is available at https://github.com/IAIK/CoronaHeatMap.

Runtime. The runtime of our protocol is $\mathcal{O}(n_v n_o)$, i.e., it scales linearly in the number of `MatMul` evaluations. This can be seen in Figure 4 in which we summarize the runtime of the homomorphic matrix multiplication for different matrix dimensions using only one thread. For better comparability, we evaluate the different sizes with the same BFV parameter set. For real-world matrix dimensions, some added runtime has to be expected due to thread synchronization and the accumulation of the intermediate thread results.

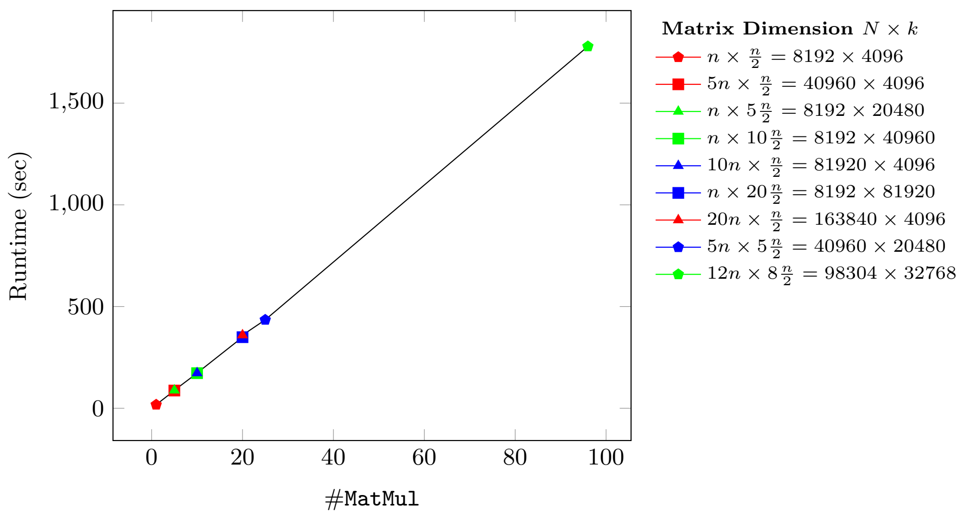

Fig. 4: Linear dependency of the runtime of the overal matrix multiplication to the number of `MatMul` evaluations. BFV parameters are: $\log_2(p) = 33$, $\log_2(q) = 218$, $n = 8192$, $\kappa = 128$.

Real World Matrix Dimension. In our benchmarks we want to evaluate our protocol with parameters suitable for smaller nation states and set the matrix dimensions to $N = 2^{23}$ and $k = 2^{15}$. This would, for example, be enough to evaluate the protocol for Austria, with a population of approximately 8.9 million people[19] and 18389 cell cites[20] at the time of writing. In Table 2 we list the runtime for a homomorphic $(1 \times 2^{23}) \times (2^{23} \times 2^{15})$ matrix multiplication, for different BFV parameters, using 88 threads. We also provide the total number of `MatMul` evaluations and the (maximum) number of evaluations per thread. We give performance numbers for BFV parameters capable of evaluating the proving

[19] https://de.statista.com/statistik/daten/studie/19292/umfrage/gesamtbevoelkerung-in-oesterreich/

[20] https://www.senderkataster.at/

mask, providing ν-bit statistical privacy against malicious clients, as well as BFV parameters only capable of evaluating the aggregation phase, which corresponds to a semi-honest privacy level.

Table 2: Runtime for the Data Aggregation Phase for different parameters using 88 threads. The column Masking indicates whether this parameter set is only able to evaluate the matrix multiplication (✗), or gives the statistical privacy ν (in bits) provided by the masking value.

	BFV				Matrix		#`MatMul`	Masking	Runtime
Nr.	$\log_2(p)$	$\log_2(q)$	n	κ	N	k	total / per thread	ν	min
1	33	218	8192	128	2^{23}	2^{15}	8192 / 96	31	59.36
2	60	218	8192	128	2^{23}	2^{15}	8192 / 96	✗	89.87
3	60	438	16384	128	2^{23}	2^{15}	2048 / 24	58	267.19
4	33	162	4096	80	2^{23}	2^{15}	32768 / 384	✗	33.55
5	33	329	8192	80	2^{23}	2^{15}	8192 / 96	31	89.32
6	60	329	8192	80	2^{23}	2^{15}	8192 / 96	58	140.82

As Table 2 shows, a matrix multiplication takes approximately 1 hour for a 33 bit plaintext modulus with $\kappa = 128$ bit computational security and 1.5 hour for the bigger 60 bit modulus. The noise budget for $n = 8192$ and a 60 bit modulus, however, is not sufficient to evaluate the masking value, which has a bigger multiplicative depth than the matrix multiplication. Increasing n leads to a performance drop, more specifically, the evaluation with a 60 bit plaintext modulus takes 4.5 hours.

Reducing the computational security level to $\kappa = 80$ bit allows us to use a smaller n for the evaluation of the mask with a 60 bit prime, and the matrix multiplication with a 33 bit plaintext modulus, splitting the respective runtimes in half. Unfortunately, n can not be reduced for the 33 bit prime with 80 bit security when masking is applied, increasing the runtime of the matrix multiplication compared to the $\kappa = 128$ bit security case. This is due to the fact that in the 80 bit security case q is composed of more distinct primes q_i. We recommend, therefore, to always use 128 bit computational security parameters for the 33 bit prime when the masking value has to be evaluated.

Data Transmission. In Table 3, we list the sizes of all the data, which has to be transmitted between the server and the client. Each row corresponds to a different parameter set from Table 2. The sizes were obtained by storing each of the described elements on the file system on the benchmarking platform. The table lists the size of the ciphertexts (ct), Galois keys (gk), and relinearization keys (rk). Galois keys are required to perform homomorphic rotations, each rotation index requires one Galois key, plus an additional key for rotating the columns. When using the baby-step giant-step algorithm, we need a key for the index 1 to calculate $\texttt{rot}(\boldsymbol{x}, j)$, and a key for the indices $k \cdot m_1$, $\forall 0 < k < m_2$. Furthermore, when masking is applied, we need the keys for the power-of-2 indices to calculate the inner product of two ciphertexts. The relinearization key is required to linearize the result of a ciphertext-ciphertext multiplication. Since we only have to perform such a multiplication when we calculate the masking values, we can omit to send the relinearization key when the mask is not applied.

In addition to the values described in Table 3, the client has to announce the used BFV parameters and the hamming weight of the input vector. These values have a combined size of less than 300 bytes.

Table 3: Data transmission in MiB for the different parameters in Table 2. Values include keys for evaluating the masking value when applicable.

	Client				Server	Total
Nr.	ct	gk	rk	Total	ct	
1	256.2	87.6	1.3	345.1	1.0	346.1
2	256.2	81.4	-	337.6	2.0	339.6
3	512.1	639.2	9.0	1160.3	2.0	1162.3
4	128.3	6.2	-	134.5	1.0	135.5
5	384.2	183.9	2.6	570.7	1.0	571.7
6	384.2	183.9	2.6	570.7	2.0	572.7

As Table 3 shows, client-to-server communication is significantly more extensive than the response of the server. The main parts of the communication are the initial ciphertexts; however, especially when masking has to be applied, the Galois keys have a significant size as well. The plaintext modulus p has little to no effect on the number of bytes, contrary to the reduction polynomial degree n, which influences the communication cost significantly. The response of the server is very small in comparison to the ciphertexts he receives from the client. One reason for that is the small parameter k compared to N. The other reason is, that our implementation performs a so-called modulus-switch to level 0 after the computation, reducing the ciphertext modulus q to only one of the moduli q_i it is composed of.

7.7 Price Estimation for Deployment in Larger Countries

In this section, we want to give an estimate of the costs of deploying our system to create a corona heatmap for a larger country, more specifically, for Germany. At the time of writing, about 83.1 million people live in Germany[21], and a total of 74280 cell sites are deployed[22]. With the BFV parameters of entry Nr. 1 in

[21] https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Bevoelkerung/Bevoelkerungsstand/_inhalt.html

[22] https://www.informationszentrum-mobilfunk.de/artikel/statistik-zur-zahl-der-funkanlagenstandorte-in-deutschland

Table 2, i.e., $n = 8192$, this corresponds to a total number of of $n_v \cdot n_o = 192755$ `MatMul` evaluations. To get 96 `MatMul` evaluations per thread, we would have to acquire 21 CPU's capable of handling 96 threads each. According to the current market prizes[23] the cost of one CPU capable of handling 96 threads is $\sim 1.5\,\$$ per hour. Taking an additional overhead by handling so many threads and combining intermediate results, we estimate the cost of evaluating the homomorphic matrix multiplication for the German use case using AWS to $\sim 60\,\$$. While noting that a trivial outsourcing of such computations is not part of our proposal, this estimate still shows that it is likely very feasible to create a heatmap once a day to gain valuable insight into the spread of the disease, even for larger countries.

8 Conclusion

Our solution shows that privacy-preserving health data analytics is possible even on a national scale. We achieved this by combining three PETs. Each of them has their known limitations, but filtering out their strengths and apply them purposefully lead to a real-world cryptographic protocol.

We are now going to discuss considerations for an actual roll-out. It is important that we only guarantee privacy as long as the health authority does not share the heatmap (outcome of protocol) with the mobile operator. There are also parameters of our system that need to be chosen in view of a particular dataset, potentially in coordination with data protection authorities, such as fixing the minimum number of aggregated individuals and differential privacy parameters.

Legal Considerations

Taking into account the nature of activities performed by the Health Authority and the electronic communication provider, the use case carried out by the TU Graz University configure an articulate situation. Both entities of our use case study should be considered data controller in relation to the raw data sets they independently manage. As a matter of fact they remain in possession of the decryption keys to anonymise datasets, but in such specific activity and would be possible for them, through deanonymisation process to turn anonymised data into personal data.

Notwithstanding such an initial situation, it is fair to assume that in the given use case, the Austrian Health Authority exercise a factual influence over the processing operation, by virtue of an exercise of decision-making power.[24] Considering this, the authority should be considered as the data controller of the whole process activity carried out in the context of this use case. The electronic

[23] https://aws.amazon.com/ec2/spot/pricing/

[24] European Data Protection Board, Guidelines 07/2020 on the concepts of controller and processor in the GDPR, adopted on 2 September 2020, https://edps.europa.eu/sites/edp/files/publication/19-11-07_edps_guidelines_on_controller_processor_and_jc_reg_2018_1725_en.pdf, p.7

service provider does not enter into possession of the decryption keys of the data sets held by the Austrian authority. Therefore, the activity performed should be considered carried out on anonymised data, so out of the EU privacy and data protection framework. If data sent by the Health Authority do not meet the criteria listed by the WP29 and recent EU jurisprudence, the electronic service provider should be considered as a mere processor, with limited security obligation.

Acknowledgments

This work was supported by EU's Horizon 2020 project Safe-DEED, grant agreement n°825225, and by the "DDAI" COMET Module within the COMET – Competence Centers for Excellent Technologies Programme, funded by the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry for Digital and Economic Affairs (bmdw), the Austrian Research Promotion Agency (FFG), the province of Styria (SFG) and partners from industry and academia. The COMET Programme is managed by FFG.

References

1. AISEC, F.: Pandemic contact tracing apps: Dp-3t, pepp-pt ntk, and robert from a privacy perspective. Cryptology ePrint Archive, Report 2020/489 (2020), https://eprint.iacr.org/2020/489
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Mathematical Cryptology **9**(3), 169–203 (2015)
3. Apple: Differential privacy. https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf (2020)
4. Bengtsson, L., Gaudart, J., Lu, X., Moore, S., Wetter, E., Sallah, K., Rebaudet, S., Piarroux, R.: Using mobile phone data to predict the spatial spread of cholera. Scientific reports **5**, 8923 (2015)
5. Berke, A., Bakker, M., Vepakomma, P., Larson, K., Pentland, A.S.: Assessing disease exposure risk with location data: A proposal for cryptographic preservation of privacy (2020)
6. Beskorovajnov, W., Dörre, F., Hartung, G., Koch, A., Müller-Quade, J., Strufe, T.: Contra corona: Contact tracing against the coronavirus by bridging the centralized–decentralized divide for stronger privacy. Cryptology ePrint Archive, Report 2020/505 (2020), https://eprint.iacr.org/2020/505
7. Brack, S., Reichert, L., Scheuermann, B.: Decentralized contact tracing using a dht and blind signatures. Cryptology ePrint Archive, Report 2020/398 (2020), https://eprint.iacr.org/2020/398
8. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: CRYPTO. Lecture Notes in Computer Science, vol. 7417, pp. 868–886. Springer (2012)
9. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: ITCS. pp. 309–325. ACM (2012)
10. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society (2018)

11. Bureau, U.C.: Statistical safeguards. https://www.census.gov/about/policies/privacy/statistical_safeguards.html (2020)
12. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: FOCS. pp. 136–145. IEEE (2001)
13. Canetti, R., Trachtenberg, A., Varia, M.: Anonymous collocation discovery: Harnessing privacy to tame the coronavirus (2020)
14. Chan, J., Foster, D., Gollakota, S., Horvitz, E., Jaeger, J., Kakade, S., Kohno, T., Langford, J., Larson, J., Singanamalla, S., Sunshine, J., Tessaro, S.: Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing (2020)
15. Chen, H., Huang, Z., Laine, K., Rindal, P.: Labeled PSI from fully homomorphic encryption with malicious security. In: ACM Conference on Computer and Communications Security. pp. 1223–1237. ACM (2018)
16. Cheon, J.H., Kim, A., Kim, M., Song, Y.S.: Homomorphic encryption for arithmetic of approximate numbers. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 10624, pp. 409–437. Springer (2017)
17. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 10031, pp. 3–33 (2016)
18. Damgård, I., Keller, M., Larraia, E., Pastro, V., Scholl, P., Smart, N.P.: Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In: ESORICS. LNCS, vol. 8134, pp. 1–18. Springer (2013)
19. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: CRYPTO. LNCS, vol. 7417, pp. 643–662. Springer (2012)
20. Dar, A.B., Lone, A.H., Zahoor, S., Khan, A.A., Naaz, R.: Applicability of mobile contact tracing in fighting pandemic (covid-19): Issues, challenges and solutions. Cryptology ePrint Archive, Report 2020/484 (2020), https://eprint.iacr.org/2020/484
21. Dwork, C.: Differential privacy. In: ICALP (2). Lecture Notes in Computer Science, vol. 4052, pp. 1–12. Springer (2006)
22. Evans, D., Kolesnikov, V., Rosulek, M.: A pragmatic introduction to secure multiparty computation. Found. Trends Priv. Secur. **2**(2-3), 70–246 (2018)
23. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive **2012**, 144 (2012)
24. Ferguson, N.M., Cummings, D.A., Cauchemez, S., Fraser, C., Riley, S., Meeyai, A., Iamsirithaworn, S., Burke, D.S.: Strategies for containing an emerging influenza pandemic in southeast asia. Nature **437**(7056), 209–214 (2005)
25. Finger, F., Genolet, T., Mari, L., de Magny, G.C., Manga, N.M., Rinaldo, A., Bertuzzo, E.: Mobile phone data highlights the role of mass gatherings in the spreading of cholera outbreaks. Proceedings of the National Academy of Sciences **113**(23), 6421–6426 (2016)
26. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC. pp. 169–178. ACM (2009)
27. Goldreich, O.: The Foundations of Cryptography - Volume 2: Basic Applications. Cambridge University Press (2004)
28. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: STOC. pp. 291–304. ACM (1985)
29. Google: Learning statistics with privacy, aided by the flip of a coin. https://ai.googleblog.com/2014/10/learning-statistics-with-privacy-aided.html (2014)
30. Google, Apple: Apple and google's exposure notification system. https://www.apple.com/covid19/contacttracing (2020)

31. Grenfell, B.T., Bjørnstad, O.N., Kappey, J.: Travelling waves and spatial hierarchies in measles epidemics. Nature **414**(6865), 716–723 (2001)
32. Halevi, S., Shoup, V.: Algorithms in helib. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 8616, pp. 554–571. Springer (2014)
33. Halevi, S., Shoup, V.: Bootstrapping for helib. In: EUROCRYPT (1). Lecture Notes in Computer Science, vol. 9056, pp. 641–670. Springer (2015)
34. Halevi, S., Shoup, V.: Faster homomorphic linear transformations in helib. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 10991, pp. 93–120. Springer (2018)
35. Hazay, C., Lindell, Y.: Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. In: TCC. Lecture Notes in Computer Science, vol. 4948, pp. 155–175. Springer (2008)
36. Isdory, A., Mureithi, E.W., Sumpter, D.J.: The impact of human mobility on hiv transmission in kenya. PloS one **10**(11), e0142805 (2015)
37. Keller, M.: MP-SPDZ: A versatile framework for multi-party computation. IACR Cryptol. ePrint Arch. **2020**, 521 (2020), https://eprint.iacr.org/2020/521
38. Krumm, J.: A survey of computational location privacy. Pers. Ubiquitous Comput. **13**(6), 391–399 (2009)
39. Lai, J., Deng, R.H., Pang, H., Weng, J.: Verifiable computation on outsourced encrypted data. In: ESORICS (1). Lecture Notes in Computer Science, vol. 8712, pp. 273–291. Springer (2014)
40. Lindell, Y., Pinkas, B.: Secure two-party computation via cut-and-choose oblivious transfer. In: TCC. Lecture Notes in Computer Science, vol. 6597, pp. 329–346. Springer (2011)
41. Lindell, Y., Pinkas, B.: Secure two-party computation via cut-and-choose oblivious transfer. J. Cryptol. **25**(4), 680–722 (2012)
42. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 6110, pp. 1–23. Springer (2010)
43. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: SODA. pp. 448–457. ACM/SIAM (2001)
44. Nissim, K., Bembenek, A., Wood, A., Bun, M., Gaboardi, M., Gasser, U., O'Brien, D.R., Steinke, T., Vadhan, S.: Bridging the gap between computer science and legal approaches to privacy. Harv. JL & Tech. **31**, 687 (2017)
45. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 1592, pp. 223–238. Springer (1999)
46. Pinkas, B., Ronen, E.: Hashomer - a proposal for a privacy-preserving bluetooth based contact tracing scheme for hamagen. https://github.com/eyalr0/HashomerCryptoRef/blob/master/documents/hashomer.pdf (2020)
47. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC. pp. 84–93. ACM (2005)
48. Rivest, R.L., Adleman, L., Dertouzos, M.L.: On data banks and privacy homomorphisms. Foundations of Secure Computation, Academia Press pp. 169–179 (1978)
49. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978)
50. Microsoft SEAL (release 3.4). https://github.com/Microsoft/SEAL (Oct 2019), microsoft Research, Redmond, WA.
51. Tatem, A.J., Huang, Z., Narib, C., Kumar, U., Kandula, D., Pindolia, D.K., Smith, D.L., Cohen, J.M., Graupe, B., Uusiku, P., et al.: Integrating rapid risk map-

ping and mobile phone call record data for strategic malaria elimination planning. Malaria journal **13**(1), 52 (2014)
52. Tatem, A.J., Qiu, Y., Smith, D.L., Sabot, O., Ali, A.S., Moonen, B.: The use of mobile phone data for the estimation of the travel patterns and imported plasmodium falciparum rates among zanzibar residents. Malaria journal **8**(1), 287 (2009)
53. Team, D.T.: Decentralized privacy-preserving proximity tracing. https://github.com/DP-3T (2020)
54. Trieu, N., Shehata, K., Saxena, P., Shokri, R., Song, D.: Epione: Lightweight contact tracing with strong privacy (2020)
55. Wesolowski, A., Buckee, C.O., Engø-Monsen, K., Metcalf, C.J.E.: Connecting mobility to infectious diseases: the promise and limits of mobile phone data. The Journal of infectious diseases **214**(suppl_4), S414–S420 (2016)
56. Wesolowski, A., Eagle, N., Noor, A.M., Snow, R.W., Buckee, C.O.: The impact of biases in mobile phone ownership on estimates of human mobility. Journal of the Royal Society Interface **10**(81), 20120986 (2013)
57. Wesolowski, A., Eagle, N., Tatem, A.J., Smith, D.L., Noor, A.M., Snow, R.W., Buckee, C.O.: Quantifying the impact of human mobility on malaria. Science **338**(6104), 267–270 (2012)
58. Wesolowski, A., Metcalf, C., Eagle, N., Kombich, J., Grenfell, B.T., Bjørnstad, O.N., Lessler, J., Tatem, A.J., Buckee, C.O.: Quantifying seasonal population fluxes driving rubella transmission dynamics using mobile phone data. Proceedings of the National Academy of Sciences **112**(35), 11114–11119 (2015)
59. Wesolowski, A., Qureshi, T., Boni, M.F., Sundsøy, P.R., Johansson, M.A., Rasheed, S.B., Engø-Monsen, K., Buckee, C.O.: Impact of human mobility on the emergence of dengue epidemics in pakistan. Proceedings of the National Academy of Sciences **112**(38), 11887–11892 (2015)
60. Xu, F., Tu, Z., Li, Y., Zhang, P., Fu, X., Jin, D.: Trajectory recovery from ash: User privacy is NOT preserved in aggregated mobility data. In: WWW. pp. 1241–1250. ACM (2017)

A Legal Aspects

Social context

Data-driven solutions are providing fundamental supports to public authorities in their fight against COVID-19. Due to the implications, such solutions have on the privacy and data protection of citizens, compliance with the EU privacy and data protection framework should be assessed. As a matter of fact. From an ethical and socio-economic perspective, should be considered the alignment of these solutions with the EU framework as a precondition to enhance citizens' trust, necessary for an efficient use of such technological novelties.

From a Member State perspective should be stressed that regardless the peculiarities of the adopted solutions, general principles of effectiveness, proportionality and necessity should always be promoted, and the adoption of any technological solutions by the public authorities should avoid any unjustified compression of the privacy and data protection of citizens.[25] Therefore, due to the large-scale processing activities and multiple actors involved in the process, a risk-based analysis should be not only desirable but also auspicate.

Legal framework

In the privacy and data protection context two central legislations should be taken into account, namely, the General Data Protection Regulation[26] and ePrivacy Directive,[27] lex specialis that exclusively Deals With '*The Processing Of Personal Data In Connection With The Provision Of Publicly Available electronic communication services in public communications networks in the community'*.[27]

The procedure developed by TU Graz uses location data provided by electronic communication servers providers to support an efficient response to the pandemic by modelling the spread of the virus through a heatmap, consequently giving the possibility to involved public authorities to develop a confinement measures.

In our case study, the processing activity involves data that falls into the definition of traffic and location data, and both defined and regulated by the Art.

[25] EDPB, 'Guidelines 04/2020 on the Use of Location Data and Contact Tracing Tools in the Context of the COVID-19 Outbreak' (2020) https://edpb.europa.eu/sites/edpb/files/files/file1/edpb_guidelines_20200420_contact_tracing_covid_with_annex_en.pdf accessed 19 October 2020.

[26] Regulation (EU) 2016/679 of The European Parliament And Of The Council on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)

[27] Directive 2002/58/EC Of The European Parliament And Of The Council Of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications), OJ L 201, 31.7.2002, p. 37

6 and 9 ePrivacy Directive. According to such provisions, traffic data and data indicating the geographic position of the terminal equipment of a user should be processed for a specific purpose and then erased or made anonymous if the consent of the user was not gained. If such information is stored on the device of the user, Art.5(3) on the confidentiality principle requires that such processing activity (access to personal data) is only allowed when authorised by the user. Nonetheless, Art 15 ePrivacy Directive derogates Art.9 restriction when *'such restriction constitutes a necessary, appropriate and proportionate measure within a democratic society to safeguard national security (i.e. State security), defence, public security, and the prevention, investigation, detection and prosecution of criminal offences or unauthorised use of the electronic communication system'*.[28] The use of such data, regardless if they can be defined as anonymised or not after the use of cryptographic techniques developed by TU Graz, falling in the scope of Art.15, seems to offer an exception for the use of such data.

GDPR and ePrivacy Scope of Application

To assess whether or not the processed data fall into the scope of application of the GDPR or ePrivacy Directive, a preliminary assessment on the data is necessary. According to Rec 26 GDPR *'the principles of data protection should not apply to anonymous information, namely, information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable'*.[29] Notwithstanding the reference to anonymised data of Rec. 26, neither the GDPR nor the ePrivacy Directive provide a definition of anonymised data. Contrary, the GDPR provides a definition of personal data and pseudonymised data. According to Art.4 GDPR, personal data are data should be considered *'any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person'*.[30] Considering such definition if data processed in the context of developing a heat map for monitoring COVID-19 positive patients data can be directly or indirectly identifiable, EU privacy and data protection requirements apply. As a result, a legal assessment of data processing activities should be subject to a case-by-case assessment.[31]

Due also to the absence of a proper definition of anonymised data in the GDPR's articles this term is often mistaken for pseudonymisation. Pseudonymisation is defined by Art 4(5) as *'the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information*

[28] Art.15 ePrivacy Directive
[29] Rec.26 GDPR
[30] Art.4(1) GDPR
[31] EDPB (n 1).

is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person'. Pseudonymisation can be done in a retraceable or untraceable way. In the former case, individuals can be identified, and consequently, these pseudonymised data fall into the scope of GDPR's scope of application. In the latter case, the process creates anonymised data and is un-retraceable, in the sense that the identity of the subject is cannot be discovered or even deleted.

Therefore, the possibility to identify the subject marks the difference between pseudonymisation and anonymisation process and consequently, the application or not of the EU privacy and data protection framework.

EU Court of Justice approach

In the context of our use case, it is crucial to mention a crucial decision made by the European Court of Justice on whether or not dynamic IP addresses can be considered as personal data (Breyer case).[32] In such a case the CJEU pronounced on the interpretation of data subject's identifiability in the Directive 95/46[33] (replaced by the GDPR). According to the Luxembourg judges the wording used in and transposed in the GDPR referring to the possibility to identify personal data by 'any other person' suggests that for information to be treated as 'personal data' it is not required that *'all the information enabling the identification of the data subject must be in the hands of one person'*.[34] Nonetheless, the Luxemburg judges, endorsing the Advocate General approach add that to identify specific data as personal data should be assessed whether it would be possible to combine data held by the data controller with means likely reasonably to be used by third parties to identify the data subject.[35]

[32] Patrick Breyer v Bundesrepublik Deutschland [2016] European Court of Justice Case C-582/14, ECLI:EU:C:2016:779 [46]

[33] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such dataOJ L 281, 23.11.1995, p. 31–50

[34] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such dataOJ L 281, 23.11.1995, p. 31–50, paragraph 43

[35] " Just as recital 26 refers not to any means which may be used by the controller (in this case, the provider of services on the Internet), but only to those that it is likely 'reasonably' to use, the legislature must also be understood as referring to 'third parties' who, also in a reasonable manner, may be approached by a controller seeking to obtain additional data for the purpose of identification This will not occur when contact with those third parties is, in fact, very costly in human and economic terms, or practically impossible or prohibited by law. Otherwise, as noted earlier, it would be virtually impossible to discriminate between the various means, since it would always be possible to imagine the hypothetical contingency of a third party who, no matter how inaccessible to the provider of services on the Internet, could — now or in the future — have additional relevant data to assist in the identification of a user" .Opinion Of Advocate General Campos Sánchez-Bordona,

B Security Proofs

Throughout this section, we will use the definitions from Section 4.5, in particular from Figure 2. Furhter, we denote the computational security parameter by κ, and the statistical security parameter by ν.

We are now going to briefly summarize what it means that a protocol is secure in the real-ideal-paradigm sense [22]. First, a protocol only can be proven secure with respect to an ideal functionality. In other words, a protocol execution is secure if it behaves the same as when the parties send their input to a trusted third party that does the computation and provides them with the outputs. More formally, an environment should not be able to distinguish between observation of the protocol with a possible adversary and a simulator interacting with the ideal functionality. More specifically, most of the time, computational indistinguishability is required between the ideal and real world. In contrast, we require (κ, ν)-indistinguishability [40] respectively [41] to analyse the cheating probability more thoroughly.

Definition 1 ([40]). *Let $X = \{X(a, \kappa, \nu)\}_{\kappa,\nu,\in\mathbb{N},a\in\{0,1\}^*}$ and $Y = \{Y(a, \kappa, \nu)\}_{\kappa,\nu,\in\mathbb{N},a\in\{0,1\}^*}$ be probability ensembles, so that for any $\kappa, \nu \in \mathbb{N}$ the distribution $\{X(a, \kappa, \nu)\}$ (resp. $\{Y(a, \kappa, \nu)\}$) ranges over strings of length polynomial in $\kappa+\nu$. We say that the ensembles are (κ, ν)-indistinguishable if for every polynomial-time adversary $\mathcal{A}$, it holds that for every $a \in \{0, 1\}^*$*

$$|\Pr[\mathcal{A}(\{X(a, \kappa, \nu) = 1\})] - \Pr[\mathcal{A}(\{Y(a, \kappa, \nu) = 1\})]| < \frac{1}{p(\kappa)} + 2^{-\mathcal{O}(\nu)},$$

for every $\nu \in \mathbb{N}$, every polynomial $p(\cdot)$, and all large enough $\kappa \in \mathbb{N}$.

B.1 Masks

Lemma 2. *Let t be a integer of bit-length $\nu \in \mathbb{N}$, and let $N \leq 2^{\nu/2}$. Further, let $\boldsymbol{x}$ and $\mu_{\texttt{bin}}$ be defined as in Section 4.3, then it holds that*

$$\Pr[\boldsymbol{x}\ \texttt{not binary}\ \wedge\ \mu_{\texttt{bin}} = 0] = \Pr\left[\boldsymbol{x} \notin \mathbb{Z}_2^N \wedge \mu_{\texttt{bin}} = 0\right] \leq \frac{1}{2^{\nu-1}}.$$

Proof.

$$\begin{aligned}\mu_{\texttt{bin}} &= \underbrace{\langle \boldsymbol{x}, (\boldsymbol{d} \circ \boldsymbol{y_1^N})\rangle \cdot r_1}_{:=\alpha} + \underbrace{\langle \boldsymbol{x}, (\boldsymbol{d} \circ \boldsymbol{y_2^N})\rangle \cdot r_2}_{:=\beta} \\ &= \alpha + \beta\end{aligned}$$

We are now interested in "bad events", i.e., when $\boldsymbol{x} \notin \mathbb{Z}_2^N$ but the binary mask is still 0. On a high-level this can only happen in two ways. Either $\alpha = \beta = 0$ or $\alpha = -\beta$. Next, we calculate the probability of these two cases.

Case C-582/14 Patrick Breyer V Bundesrepublik Deutschland, 12 May 2016 (1), ECLI:EU:C:2016:339, paragraph 68.

First, since $r_1, r_2 \neq 0$ and assuming $\boldsymbol{x} \neq \mathbf{0}^k$ (which is a valid input and no "bad event"), we have $\Pr[\alpha = 0] = \Pr[\beta = 0] = {}^{N}\!/t$ [10]. Hence,

$$\Pr[\alpha = \beta = 0] = \frac{N}{t} \cdot \frac{N}{t} = \frac{N^2}{t^2}. \tag{8}$$

Consequently, the probability of α being non-zero is $1 - {}^{N}\!/t$. Further, the probability of β being $-\alpha$ is ${}^{1}\!/t$. Combing these probabilities gives us

$$\Pr[\alpha = -\beta] = \left(1 - \frac{N}{t}\right)\frac{1}{t} = \frac{1}{t} - \frac{N}{t^2}. \tag{9}$$

We get the final probability by putting together Equation (8) and Equation (9)

$$\begin{aligned}\Pr[\alpha + \beta = 0] &= \frac{N^2}{t^2} + \frac{1}{t} - \frac{N}{t^2} < \frac{1}{t} + \frac{N^2}{t^2} \\ &\leq \frac{1}{2^\nu} + \frac{2^\nu}{2^{2\nu}}, \text{ because } N \leq 2^{\nu/2} \\ &= \frac{1}{2^{\nu-1}}.\end{aligned}$$

□

Corollary 1. *Let t be a integer of bit-length $\nu \in \mathbb{N}$. Further, let $N \leq 2^{\nu/2}$, and $\boldsymbol{\mu}$ be the result of Equation* (2)*, then it holds that*

$$\Pr\left[\boldsymbol{\mu} = \mathbf{0}^k \wedge (\boldsymbol{x} \notin \mathbb{Z}_2^N \vee w \neq \langle \boldsymbol{x}, \mathbf{1}^N \rangle)\right] \leq \frac{1}{2^{\nu-2}},$$

Proof. Since $\mu_{\mathtt{bin}}$ is controlled by the client, he has a chance of $1/t$ to guess and counteract a non-zero $\mu_{\mathtt{bin}}$ in $\mathbb{Z}_t$. Therefore, a vector $\boldsymbol{x}$ which is either non-binary, has a hamming weight $\neq w$, or both will result in a masking value of $\mathbf{0}^k$ only with probability $\Pr\left[\mu_{\mathtt{bin}} = 0 \wedge x \notin \mathbb{Z}_2^N\right] + 1/t$. With a ν bit t and $N \leq 2^{\nu/2}$, this probability will thus be:

$$\Pr\left[\boldsymbol{\mu} = \mathbf{0}^k \wedge (\boldsymbol{x} \notin \mathbb{Z}_2^N \vee w \neq \langle \boldsymbol{x}, \mathbf{1}^N \rangle)\right] \leq \frac{1}{2^{\nu-1}} + \frac{1}{2^\nu} = \frac{3}{2^\nu} \leq \frac{1}{2^{\nu-2}},$$

□

B.2 Proof of Lemma 1

Proof. We use Lemma 2 to prove that to any polynomial time environment the execution π_{CoV} with a possible adversary $\mathcal{A}$ is (κ, ν)-indistinguishable from a simulator $\mathcal{S}$ interacting with the ideal functionality $\mathcal{F}_{CoV}$. More concretely, we claim that as long the event that $\boldsymbol{x}$ is not binary and at the same time the mask $\boldsymbol{\mu} = \mathbf{0}^k$ does not occur, the executions of the ideal and real world are computational indistinguishable. Once we have proven this claim, we are done, since we have already shown that the probability of the above event is

π_{CoV}

1. A party P_1 on input $(\mathsf{input}, sid, P_1, P_2, \boldsymbol{x})$ from the environment verifies that $\boldsymbol{x} \in \mathbb{Z}_t^N$, else ignores the input. Next, samples a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.KGen}(1^\kappa)$ and $\boldsymbol{c} \leftarrow \mathsf{HE.Enc}_{\mathsf{pk}}(\boldsymbol{x})$. It records $(sid, P_1, P_2, \mathsf{sk})$, and sends $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$ to P_2. P_1 ignores subsequent inputs of the form $(\mathsf{input}, sid, P_1, P_2, \cdot)$ from the environment.
2. On a later input of the form $(sid, P_1, P_2, \boldsymbol{h}^*)$ from P_2, P_1 computes $\boldsymbol{h} \leftarrow \mathsf{HE.Dec}_{\mathsf{sk}}(\boldsymbol{h}^*)$, and outputs $(\mathsf{result}, sid, P_1, P_2, \boldsymbol{h})$ to the environment.
3. A party P_2 on input $(\mathsf{input}, sid, P_1, P_2, Z)$ from the environment and $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$ from P_1 verifies that $Z \in \mathbb{Z}_t^{N\times k}$, else ignores the input. Next, computes the mask vector $\boldsymbol{\mu}$ and the noise $\boldsymbol{\delta}$ according to Figure 2. Then computes $\boldsymbol{h}^* \leftarrow \mathsf{HE.Eval}_{\mathsf{pk}}(\boldsymbol{c}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu})$. P_2, sends $(sid, P_1, P_2, \boldsymbol{h}^*)$ to P_1 and ignores all subsequent inputs of the form $(\mathsf{input}, sid, P_1, P_2, \cdot)$ from the environment.

Fig. 5: Formalized protocol π_{CoV}

exponentially small in the statistical security parameter. Note that for the proof, we rewritten the protcol in a more formal description π_{CoV}, see Figure 5.

Before going into the proof of the claim let us note recall that the adversary $\mathcal{A}$ is static, i.e., the set of corrupted parties is fixed from the start and known to the simulator $\mathcal{S}_{CoV}$. Therefore it has full control of the corrupted dummy parties.

First consider a polynomial time environment which does not corrupt any of the parties. Any meaningful environment will interact with π_{CoV} or $\mathcal{F}_{CoV}$ in the following way.

1. It picks some vector $\boldsymbol{x} \in \mathbb{Z}_t^n$ and inputs $(\mathsf{input}, sid, P_1, P_2, \boldsymbol{x})$.
2. Then it sees $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$.
3. Then it picks some matrix $Z \in \mathbb{Z}_t^{N\times k}$ and inputs $(\mathsf{input}, sid, P_1, P_2, Z)$.
4. Then it sees $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{h}^*)$.
5. Then it sees $(\mathsf{result}, sid, P_1, P_2, \boldsymbol{h})$.

Let now us assume to the contrary there is such an environment $\mathcal{E}$ that can distinguish the two systems $\pi_{CoV} \circ \mathcal{A}$ and $\mathcal{F}_{CoV} \circ \mathcal{S}$ with non-negligible advantage. Then we can turn $\mathcal{E}$ into a polynomial time system $\mathcal{E}'$ which wins in the IND-CPA game with non-negligible probability:

1. First $\mathcal{E}'$ receives pk.
2. Then $\mathcal{E}'$ runs $\mathcal{E}$ to see which message $(sid, P_1, P_2, \boldsymbol{x})$ gets recorded.
3. Then $\mathcal{E}'$ inputs $(\boldsymbol{x}, \boldsymbol{0}^N)$ to the IND-CPA game and gets back an encryption $\boldsymbol{c}$, where $\boldsymbol{c}$ is either an encryption of $\boldsymbol{x}$ (if $b = 0$) or an encryption of $\boldsymbol{0}^N$ (if $b = 1$).
4. Then $\mathcal{E}'$ samples $Z \leftarrow \mathbb{Z}_t^N$. It runs $\mathcal{E}$ and provides input $(\mathsf{input}, sid, P_1, P_2, \boldsymbol{x})$, $(\mathsf{input}, sid, P_1, P_2, Z)$, $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$, $(sid, P_1, P_2, \mathsf{HE.Enc}_{\mathsf{pk}}(\boldsymbol{c}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu}))$ and $(\mathsf{result}, sid, P_1, P_2, \boldsymbol{x}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu})$.
5. $\mathcal{E}'$ waits until $\mathcal{E}$ outputs its guess b', and then $\mathcal{E}'$ outputs b'.

$\mathcal{S}_{CoV}$

P_1 and P_2 not corrupted: It starts by sampling a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.KGen}(1^\kappa)$, and sets $\boldsymbol{x} \leftarrow 0^N$. Then it computes $\boldsymbol{c} \leftarrow \mathsf{HE.Enc}_{\mathsf{pk}}(\boldsymbol{x})$, and instructs P_1 to send $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$ to P_2.
On later input of the form $(sid, P_1, P_2, \mathsf{pk}, \boldsymbol{c})$ from P_1 it samples $Z \leftarrow \mathbb{Z}_t^{N \times k}$. Then it computes $\boldsymbol{h}^* \leftarrow \mathsf{HE.Eval}_{\mathsf{pk}}(\boldsymbol{c}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu})$. It instructs P_2 to send $(sid, P_1, P_2, \boldsymbol{h}^*)$ to P_1.

P_1 not corrupted, P_2 corrupted: Similar as before but it does not have to simulate Z because it learns the input Z from P_2. Then it simply computes $\mathsf{HE.Eval}_{\mathsf{pk}}(\boldsymbol{c}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu}$.

P_1 corrupted, P_2 not corrupted: It learns the input $\boldsymbol{x}$ from P_1. Then it proceeds as in the first case until it has to simulate the message to P_1. In order to do this it runs a copy of π_{CoV} internally, where it corrupts P_1. Thereby, it learns $\boldsymbol{x}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu}$ and sets $\boldsymbol{h}^* \leftarrow \mathsf{HE.Enc}_{\mathsf{pk}}(\boldsymbol{x}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu})$.

P_1 and P_2 corrupted: It learns the inputs $\boldsymbol{x}$ from P_1 respectively Z from P_2. It simply runs the protocol with the inputs, and outputs $(\mathsf{input}, sid, P_1, P_2, \boldsymbol{x})$ and $(\mathsf{input}, sid, P_1, P_2, Z)$ to the ideal functionality, which makes $\mathcal{F}_{CoV}$ output $(\mathsf{result}, sid, P_1, P_2, \boldsymbol{x}^T \cdot Z + \boldsymbol{\delta} + \boldsymbol{\mu})$.

Fig. 6: Simulator $\mathcal{S}_{CoV}$.

If $b = 0$, then $\mathcal{E}$ observes the interaction it would see when interacting with the protocol π_{CoV}, and if $b = 1$, then $\mathcal{E}$ observes the interaction it would see when interacting with the ideal functionality and the simulator $\mathcal{F}_{CoV} \circ \mathcal{S}$. By assumption $\mathcal{E}$ can distinguish $\pi_{CoV} \circ \mathcal{A}$ and $\mathcal{F}_{CoV} \circ \mathcal{S}$ with non-negligible advantage. Therefore, $\mathcal{E}'$ will guess b with probability significantly better than $1/2$. This is a contradiction to the IND-CPA security of HE, as $\mathcal{E}'$ is polynomial time. $\square$

B.3 One-Sided Simulation

To define one-sided simulation security, we have the notion of a protocol execution view. Let $VIEW^{\mathcal{A}}_{\pi,\mathcal{A}}(x, y)$ denoted the protocol execution view of the adversary $\mathcal{A}$, i.e., the corrupted parties' view (input, randomness, all received messages) after execution of π with input x respectively y from P_1 respectively P_2.

Definition 2. *Let $EXEC_{\pi,\mathcal{A},\mathcal{E}}$ respectively $EXEC_{\mathcal{F},\mathcal{S},\mathcal{E}}$ denote the random variables describing the output of environment $\mathcal{E}$ when interacting with an adversary $\mathcal{A}$ and parties P_1, P_2 performing protocol π, respectively when interacting with a simulator $\mathcal{S}$ and an ideal functionality $\mathcal{F}$, where only P_1 receives output. Protocol π securely realizes functionality $\mathcal{F}$ with one-sided simulation if*

1. *for any adversary $\mathcal{A}$ that controls P_2 there exists a simulator $\mathcal{S}$ such that, for any environment $\mathcal{E}$ the distribution of $EXEC_{\pi,\mathcal{A},\mathcal{E}}$ and $EXEC_{\mathcal{F},\mathcal{S},\mathcal{E}}$ are indistinguishable,*

2. and for any adversary $\mathcal{A}$ controlling P_1 the distribution $VIEW^{\mathcal{A}}_{\pi,\mathcal{A}}(x, y)$ and $VIEW^{\mathcal{A}}_{\pi,\mathcal{A}}(x, y')$, where $|y| = |y'|$ are indistinguishable.

C Differential Privacy

Let us recall the definition of ϵ-differential privacy [21]:

Lemma 3 (ϵ-Differential Privacy). *A randomized mechanism $\mathcal{A}$ gives ϵ-differential privacy if for any neighboring datasets D and D', and any $S \in Range(\mathcal{A})$: $Pr[\mathcal{A}(D) = S] \leq e^{\epsilon} Pr[\mathcal{A}(D') = S]$.*

Since D and D' are interchangeable, Lemma 3 implies:

$$e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(D) = S]}{Pr[\mathcal{A}(D') = S]} \leq e^{\epsilon}$$

$$\text{i.e. for small } \epsilon : \quad 1 - \epsilon \lesssim \frac{Pr[\mathcal{A}(D) = S]}{Pr[\mathcal{A}(D') = S]} \lesssim 1 + \epsilon$$

An established technique to achieve ϵ-differential privacy is the Laplace mechanism, i.e., to add noise from a zero-centered Laplace distribution to the final result of the computation. The noise is, thereby, calibrated with the privacy budget ϵ and the global sensitivity Δq of the computation q: $\Delta q = \max_{D,D'} ||q(D) - q(D')||$ for all neighboring D and D'. In other words, the global sensitivity represents the maximum possible value of each element in the dataset. The Laplace distribution for a scale factor b is given as $Lap(x|b) = \frac{1}{2b} e^{-\frac{|x|}{b}}$, in the Laplace mechanism a scale factor of $b = \frac{\Delta q}{\epsilon}$ is used.

D BFV parameters

In this section we list the BFV parameters used in our implementation. In BFV, one can choose three different parameters which greatly impact the runtime, security, and the available noise budget (i.e. how much further noise can be introduced until decryption will fail):

– Plaintext modulus t: t defines the Ring $\mathbb{Z}_t$ to which the homomorphic operations correspond to. Every result encoded in the ciphertext vector will be an element of $\mathbb{Z}_t$. Therefore, one has to make sure that t is big enough, such that no computation overflows. On the other hand, a big t has a bad impact on the ciphertext noise, where the noise cost of homomorphic operations is higher for bigger t. Additionally, the size of t will also affect the runtime of homomorphic operations. In general, SEAL allows arbitrary plaintext moduli $t \geq 2 \in \mathbb{Z}$; however, if we want to enable SIMD-packing (Section 7.1), then the plaintext modulus has to be a prime p and congruent to 1 (mod $2n$).

- Ciphertext modulus q: q defines the available noise budget. Therefore, a bigger q allows for a bigger depth in homomorphic operations. However, bigger q's have an adverse effect on the security of the encryption scheme. Additionally, q also influences the runtime of homomorphic operations; more specifically, the number of primes q is composed of. The more primes, the longer the computation times.
- Degree n of the reduction polynomial: In BFV in SEAL n is always a power of two and has a direct impact on the runtime of the scheme. A bigger n drastically increases the time a homomorphic operation needs for evaluation. On the other hand, a bigger n also increases the security of the scheme and, therefore, allows for a bigger ciphertext modulus q to increase the noise budget.

D.1 Plaintext Moduli

In our benchmarks, we use two different plaintext moduli, one with a size of 33 bits, the other with a size of 60 bits. Table 4 lists the used moduli.

Table 4: Used plaintext moduli in hexadecimal notation and their size in bits.

Nr.	p	$\log_2(p)$
1	`0x1e21a0001`	33
2	`0xf4fc03ff53d0001`	60

D.2 Ciphertext Moduli

In this section we list all the ciphertext moduli used for different security levels κ and reduction polynomial degrees n. In SEAL the ciphertext modulus q is the product several primes q_i: $q = \prod_i q_i$.

$n = 4096$, $\kappa = 80$: The ciphertext modulus q is composed of 3 primes with a total size of 162 bit, which we list in Table 5.

Table 5: Primes composing the ciphertext modulus for $n = 4096$, $\kappa = 80$ in hexadecimal notation and their size in bits.

i	q_i	$\log_2(q_i)$
1	`0x3fffffffd6001`	54
2	`0x3fffffffd2001`	54
3	`0x3fffffffbe001`	54

$n = 8192$, $\kappa = 80$: The ciphertext modulus q is composed of 7 primes with a total size of 329 bit, which we list in Table 6.

Table 6: Primes composing the ciphertext modulus for $n = 8192$, $\kappa = 80$ in hexadecimal notation and their size in bits.

i	q_i	$\log_2(q_i)$
1	`0x7ffffffec001`	47
2	`0x7ffffffc8001`	47
3	`0x7ffffffb4001`	47
4	`0x7ffffff00001`	47
5	`0x7fffffefc001`	47
6	`0x7fffffecc001`	47
7	`0x7fffffe70001`	47

$n = 8192$, $\kappa = 128$: The ciphertext modulus q is composed of 5 primes with a total size of 218 bit, which we list in Table 7.

Table 7: Primes composing the ciphertext modulus for $n = 8192$, $\kappa = 128$ in hexadecimal notation and their size in bits.

i	q_i	$\log_2(q_i)$
1	`0x7fffffd8001`	43
2	`0x7fffffc8001`	43
3	`0xfffffffc001`	44
4	`0xffffff6c001`	44
5	`0xfffffebc001`	44

$n = 16384$, $\kappa = 128$: The ciphertext modulus q is composed of 9 primes with a total size of 438 bit, which we list in Table 8.

Table 8: Primes composing the ciphertext modulus for $n = 16384$, $\kappa = 128$ in hexadecimal notation and their size in bits.

i	q_i	$\log_2(q_i)$
1	`0xfffffffd8001`	48
2	`0xfffffffa0001`	48
3	`0xfffffff00001`	48
4	`0x1fffffff68001`	49
5	`0x1fffffff50001`	49
6	`0x1ffffffee8001`	49
7	`0x1ffffffea0001`	49
8	`0x1ffffffe88001`	49
9	`0x1ffffffe48001`	49