UNIVERSITY POLITEHNICA OF BUCHAREST
DOCTORAL SCHOOL OF AUTOMATIC CONTROL AND COMPUTERS

Summary

PhD Thesis

In Computers and Information Technology

Security and Privacy in Big Data

Iulia-Maria Florea

Thesis advisor:
Prof. dr. ing. Răzvan-Victor Rughiniș

BUCHAREST

2022

Abstract

Privacy-preserving solutions have emerged as an absolute prerequisite for exchanging sensitive information when developing data analysis and validation algorithms. Privacy focuses on respecting individuality, while the information of interest can be drawn from population's patterns. Their main drawback of these methods is their complexity, lack of documentation and lack of open-source exemplifying algorithms. On the other hand, traditional security mechanisms fail to handle big data, due to its velocity, variety and large volume.

The present thesis builds an ecosystem for a more complex discussion about the feasibility of privacy solutions in big data. When combining research-oriented solutions with various and large amounts of information, multiple research questions can be raised. One of them regards the tradeoff between data availability and security. Another one refers to the best-known cryptographic algorithms and their applicability.

Our purpose is to build a step-by-step processing system for big data, understand the need for privacy at each layer and develop ways on how the existing technologies can suit in the legal requirements and regulations without diminishing the user's contentment. We defined four major actions in data transition: gathering, transmission, storing and actual processing. For each of them, we chose several reality-based examples and we developed proof-of-concept privacy preserving applications. Our goals revolved around reaching the usability limits of research-oriented solutions, developing them for more complex scenarios and building better ways of encoding data to maximize both users' experience quality and security.

We started by robustly exploring the simplified networking stack of embedded devices, the standardized protocols, the suitable security enhancements and their limitations. We further inspected the existing encryption, authentication and authorization methods in constrained devices. Afterwards, we proposed a feasible extension of the most common IoT based scenarios with secure protocols.

Regarding data transmission, we decided to start the research from one of the fundamental concepts of security, differential privacy. We researched the state-of-the art methods of providing geo-indistinguishability and we developed a new method on hiding the exact location of mobile users in cities. We further investigated the well known attacks and we tested our implementation against them.

At the data storing layer, we focused our work on one of the most compelling privacy-oriented solutions: searchable encryption. We considered several flavors (including hidden vector encryption and inner product encryption) and we found the most appropriated scenarios for each one. Then, we proposed customized encoding methods to reduce the overhead of applying state-of-the-art searchable encryption techniques, which are notoriously expensive.

In the end, we moved our interest towards data processing and how privacy can be implemented in machine learning algorithms. Since there has been a growing interest in automation and gaining insights on data, we considered this topic as a relevant one in the present. Despite of the general interest, machine learning algorithms are resource-intensive and an extra layer of security may lead to faults or unsatisfactory results. We developed a classification algorithm using a mix between machine learning algorithms and manual labeling using an expert's point of view to demonstrate that knowledge can be obtained from any type of data. This way, we raised the importance of privacy at this level and we conducted some experiments to understand the current state of privacy-preserving solutions. We found the main bottlenecks and how they can be mitigated.

We built a complex solution for a privacy-preserving process, but in a vast domain like big data, this type of solution is not unique, nor universal and it can rather be seen as a starting point for more specialized research questions.

1 INTRODUCTION

In December 2014, a professor from Cambridge University warned the legal department of the University about an application built by a psychology professor that collected data of millions of Facebook users without their knowledge. [1] This type of data could be used to gather insights on people's personalities, behavior and needs. The main client of the application was a little-known political consulting firm, called Cambridge Analytica [1]. What happened next was one of the most important scandals of the last few years: the implications of Facebook data usage in the American elections of 2016, that led to numerous discussions on data privacy.

One outcome is the General Data Protection Regulation (GDPR), applied since May 2018 in The European Union, consisting of a set of rules on collecting, storing and processing personal data. They refer not only to direct identifiers, such as full name or national ID number, but also to indirect information such as phone numbers, IP addresses or photos. [2] Their purpose is to force the analysis to focus on statistical findings, without being able to re-identify single individuals, when comparing the anonymous dataset with other similar sets. Re-identification is a common issue in data privacy and basic anonymization techniques cannot handle insights from background knowledge. As an example, there have been some famous re-identifications in the past and one of them is based on Netflix dataset in 2006. The company started a competition to find more accurate movie recommendations methods and, as training data, they released an anonymized dataset with movie reviews from almost 500.000 customers. The personal identifiers of the customers were removed and the datasets consisted of unique subscribers' IDs, movie ratings and rankings' dates. A team of researchers inferred the provided anonymized information with the public IMDB database and was able to identify the users by comparing the movie rankings and the comments dates. GDPR considers a dataset anonymous when re-identification is unlikely or hard to be obtained. The technical aspects of anonymization are one of points covered by this research.

Apart from poor anonymization techniques, security issues have become more of a struggle in the last few years. The number of cybernetic attacks has increased substantially in the last few years and the total cost of cybercrime has had a 15% growth rate every year. It has raised from $3 trillion in 2015 to a potential $10.5 trillion in 2025. Also, the frequency of cyberattacks has increased almost 4 times, from one at each 40 seconds in 2016 to one at each 11 seconds in 2021. [3] Data privacy in this case is even more at risk, since attackers may steal the plaintext information and infer data from multiple sources.

Since the outbreak of COVID-19 pandemic, the cybercrime increased with 600% [4] and 57% of the IT decision makers in companies consider that remote workers can pose an extra threat to the internal security. Their mission is to build a privacy-driven solution and try to add some extra layers of security, if possible. The first one is to migrate data on cloud instead of

having it locally, especially to have multiple snapshot and backups. Another solution would be to ensure protection for all the data. This requires holistic solutions, to make sure that remote workers have global access to everything they need and all the information must be compliant to local regulations.

All these requirements, illustrated in Figure 1.1, lead to multiple research questions in the privacy area, a chapter that has been seen rather as academic and research oriented than functional in real-life scenarios. This has changed in 2016, when privacy oriented concepts, content-oriented, such as differential privacy [5] have been popularized by companies such as Google [6] or Apple [7].

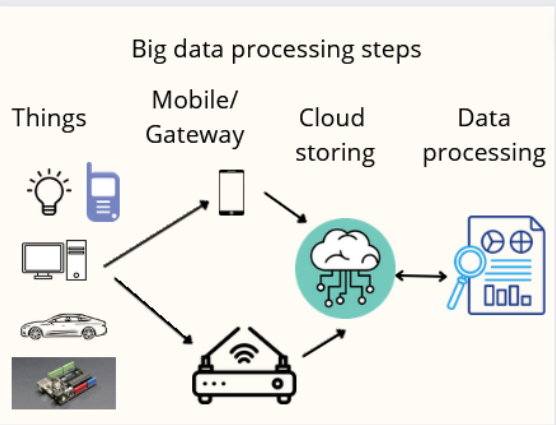

Fig. 1.1 The processing steps of big data

2 THESIS SCOPE

2.1 Research questions

The research done in this thesis follows the major data processing steps: data gathering, collecting or transmitting, storing and processing.

We began analyzing data gathering and, more specific, we considered the IoT devices a main source of new information. Research on this topic had been done for a while, so we were rather interested in understanding if the security provided by the state-of-the-art protocols was stable enough and hard to be compromised in some real-life secure scenarios. The question raised in this chapter is the following:

> **Q1**: What are the main security protocols of IoT and how can they be combined in smart-objects scenarios?

The second step focused on collecting and transmitting data from a mobile device storage server. Since IoT devices are rather stationary, we considered a different scenario, of a mobile phone and a hot topic of today's life: privacy while using location-based services. This offered a new research question:

> **Q2**: Is it feasible to provide privacy while using location-based services? Which are the limits, the main attacks and how can this be done without affecting the user's needs and experience?

The third step refers to data storage. Here, we focused on rather academic ideas, pairing based cryptography and fully homomorphic encryptions. These have been in the spotlight in the latest year, but there are still some subjects that need further research.

> **Q3**: What type of data can be stored using fully homomorphic encryption? What type of algorithm do we use for different types of information and what are the database limits for an allowed overhead?

The fourth step refers to data processing. Here, we combined two highlighted topics of today's world, artificial intelligence and privacy. We chose to build a specific classification algorithm for technical short texts and we provided some answers related to privacy in machine learning algorithms. We based our research on two different questions:

> **Q4**: Is privacy necessary in any specific scenarios? Can users get insights from any type of data, such as specialized technical texts, without annotations?

> **Q5**: What is the overhead of adding privacy in machine learning and what is the most feasible use-case for this?

2.2 Contributions

The thesis has its roots in the security and privacy issues of the world in the last few years, beginning with the increasing number of security attacks relying on IoT devices, the major scandals revolving around people's data and the increasing concern of protecting the privacy of sensitive information.

The first step in the research was to understand the most prominent protocols of IoT, at each level of the specific network stack. We discovered a simplified network stack that contains physical, MAC, network, adaptation, transport and application layers. We found the existing layers of security of each level and focused on studying security and interconnection of the application layer protocols. In this research, we have found the answer of the first research question:

> **A1**: The work in [9] and [10] proves that the relevant security protocols at the application layer are DTLS and symmetric key encryption, which can be part of a smart-campus scenario, focusing on monitoring the resources consumption.

We reached the conclusion that this topic has been widely explored lately and that the existing methods, if implemented, are enough to provide the necessary layer of security.

On the other hand, at data collection and transmission step, we found privacy related topic that had not been explored enough and was rather still in the research area. We chose to study the privacy of the mobile users who send data to HTML5 enabled websites that got location information. We started from the assumption that the exact location was not always necessary and the privacy risks may sometimes be higher than the benefits, since location data can lead to insights on personal hobbies, schedule or health issues. We followed the second question in the previous section and we found that:

> **A2**: It is feasible to add noise to an exact location, even to build a false, but realistic path in a city, without high overhead. The work in [11] and [12] show the type of algorithm that can be used, the scenarios where it can be implemented, the necessary amount of storage space and processing power and the main types of attacks that need to be avoided.

The next contributions focused on the storage step and combined a research and academic topic, searchable encryption, with big data. We started from different state-of-the-art algorithms, extended and tailored them for a specific usecase. Also, we conducted some experiments on how the existing algorithm would work on different types of data. Searching through encrypted data without needing to decrypt it would solve an important privacy issue. We have followed the third research question and we have provided the following answer:

> **A3**: Fully homomorphic encryption can be firstly tested on numerical data. It is possible to extend algorithms that only find data with some properties, to also decrypt it. The

overhead can be reduced by encoding data in smaller structures and, depending on the data owner and users, faster schemes with symmetric keys can be used. The work in [13] details how encoding and different types of searches can be done, while the work in [14] focuses on finding the most suitable algorithm for different types of numerical data.

After data is retrieved and stored, it can be used for processing. Most machine learning algorithms require training data to be annotated, but we were interested in how data without annotations can be processed. The main purpose of data privacy is to avoid obtaining personal information on people without their knowledge and we were interested to find out if this kind of scenario would be likely. We started by building a proof on concept, a classification algorithm for short technical text and we followed the privacy question also raised before. We got to the following answers:

A4: The work in [15] proves that, in some cases, unsupervised classification algorithms fail to provide good results. We ran them over short technical texts and the results were not satisfying. Nevertheless, they can be successfully integrated with some manual categories defined by an expert and provide interesting results. This proves that most types of data can offer valuable information and privacy should be a concern.

A5: The work in [16] showed the state-of-the-art privacy preserving machine learning algorithms can be integrated in classification. We discovered that the main overhead happens when transmitting and receiving data between a central server and private workers, that these algorithms are almost as accurate as the centralized solutions and that they are feasible to run on a high number of workers such as smartphones, while charging.

3 GATHERING DATA

The Internet of Things (IoT) has become an important research field. It can be described as a communication system where any device can be connected to the Internet and be able to identify itself to other objects. It can be applied in different domains, from personal use, such as smart homes or wearable devices to fields such as urban environment monitoring, health care, industrial automation or emergencies.

Generally, the IoT devices have low memory, reduced battery capacity, reduced processing capabilities and vulnerable radio conditions. The standard TCP/IP stack is not suitable for this environment, so working groups have started to adapt the existing protocols to new versions for IoT.

An addressing scheme, such as IPv6 should be taken into consideration, since there are billions of interconnected nodes. Multiple working groups already started to standardize IoT specific protocols, such as 6LoWPAN [21](RFC 4944 and RFC 6282)[25], IEEE802.15.4 [23] and ZigBee describe ways of enabling IPv6 in constrained environments. Other requirements refer to security and privacy, since the number of Denial of Service attacks has recently increased.

At the application layer, a common way of retrieving and requesting data is using Web architecture, and more specific, HTTP. This uses URIs as resource identifiers and it is based on REST architecture to publish information. For embedded devices, there is a Constrained RESTful Environments (CoRE) IETF working group that aims to develop RESTful protocols, compatible with HTTP for resource-constrained devices. They specified CoAP[34], an application-layer protocol for IoT.

3.1 Proposed Smart Campus Scenario

A proposed idea of a Smart Campus scenario focused on monitoring the comfort of students and professors in the classrooms and offices of university buildings. We proposed the deployment of an IoT network composed of gateway devices and a large number of sensor nodes.

The purpose of this IoT network was to monitor ambient parameters such as temperature, humidity, pressure, luminosity, air quality and presence. The collected data was analysed using various algorithms to detect events that would bring discomfort to students and professors, for example: low luminosity or low temperature during classes, high concentration of CO2, etc. In case of such events, the system may adjust ambient parameters, for example regulate temperature through the air conditioning system.

As any system, a smart campus should be secured. Data cannot flow in plain text between devices, it should be confidential and private. The first step we focused on was to add security at

the device layer, to encrypt communication between the nodes and the storage servers. At this level, the devices should prove their identity, prevent unauthorized access, sign and encrypt data. In IoT networks, the gateway also translates data between different wireless protocols, since 802.11 is not a common solution for IoT devices. This should also be taken into consideration, since it is responsible to maintain data integer and private when translating between protocols.

For authentication, X.509 certificates cannot be used, since the IoT devices do not have enough memory and processing power to validate them. Instead, radio-frequency identification, shared secret or MAC addresses can be used. DTLS is one of the most common security protocols, based on TLS and providing privacy and encryption. Security in CoAP [34], at application layer, is often based on DTLS [39]. It is the UDP based version of TLS, designed to provide end-to-end security. It provides flexible negotiation, using cipher suits and cryptographic mechanisms. The impact on constrained devices is related to the initial handshake and to processing all the secured packets.

3.2 Insights

The necessary security protocols for IoT devices are already developed and work should focus on enabling them on constrained devices. Based on DTLS, customized operations can be performed. We proposed a lightweight solution, using pre-shared keys for client-server architectures. Embedded hardware supports key derivation with AES-128 and, in DTLS [39], two options 'AUTH' and 'AUTH_MSG_TYPE' were added. Every client had a key configured and the server had a pool for each possible client id. The first message sent from the client to the server contained the client id and a unique token. The server found the password associated with the device, derived the key, added a random variable *nonce_1* and sent an encrypted message as a challenge. When the client got the challenge, it used the shared key to decrypt the packet, got the key and the value of nonce_1 and sent a response using the derived key and the token. If the token was the same as in the initial transmission, then the client was authenticated. We encrypted the application layer payload and used the options in CoAP header to separate encrypted from plain-text queries.

Learning how to develop a smart system for buildings, campuses and homes leads to gaining knowledge on both the advantages and issues related to connection multiple and heterogeneous devices. On one hand, they can improve the quality of life and resource usage, but they also come with security and privacy risks. Adding security to IoT is a challenging task, since the most secure mechanisms in the TCP/IP stack are not suitable. Also here, the standard DTLS protocol can be used, but together with improvements that lead to less overhead, especially at the initial handshake.

4 COLLECTING AND SENDING DATA

Location hiding is addressing the subject of location privacy provide solutions based on several scientific terms borrowed from similar subjects related to privacy, such as [50] and [51].

The term *k-anonymity* [52] is often used in the context of location privacy. Initially, it was related to the area of data management and statistics. Having a data set containing a list of persons and identifying information about each of them, *k-anonymity* is a property of a subset of this data set, which states that any combination of data in this subset will no longer identify less than *k* people in the subset while still maintaining the usefulness of the data. Through a generalization algorithm, this aims to preserve relevant data in the data set, while hiding specific information. Later, the concept was adopted in the location privacy domain also. For instance, we can assume having a map with identical anonymous dots representing locations of users and a data set containing the names and the GPS coordinates of every user. By matching the information in the data set with the map, we would be able to identify every single user. *K-anonymity* algorithm transforms the GPS locations into more general values, such as the street, the neighborhood, the city, the country the users are in. This way, when trying to match the two sources of data, an attacker will be able to tell that *k* users are in the same city, but he will not be able to individually match each user to his location on the map.

Another concept is *differential privacy* [80]. This concept was introduced in the field of statistics and it makes use of randomization and noise in order to keep data anonymous. When a query is applied on two adjacent databases (that have only one row different), it must return similar results. [10] provides a formal definition of differential privacy. A randomized function *K* provides differential privacy if, for any two datasets D_1 and D_2 that differ on a single row and all $S \subset Range(K)$:

$$Pr[(K(D_1) \epsilon S] \leq exp(\epsilon) \times Pr[(K(D_2) \epsilon S] \qquad (4.1)$$

The definition above claims that, when adding a new set to a database, the information should affect the result of a randomized function applied over that database to a certain level, less than $exp(\epsilon)$. A higher value of ϵ implies less privacy and less noise added to data, while a lower value of ϵ implies more privacy and more quality loss. The differential privacy techniques take the input from the user and alter it by applying a distribution function, such as Laplace distribution. The output of this function will be similar to the input to a certain degree, in order to still be useful for the statistical product. However, an attacker is unable to reconstruct the initial input starting from the output, therefore preserving user anonymity.

In location privacy, adding noise to a precise location gives a result that may accurately place a user within a desired geographical area, while hiding the exact location in that area. This is useful in many scenarios, a common example being a weather service. The service needs a

general location (city, part of a city) to provide accurate weather data, but it does not require a street-level location accuracy. The user receives weather forecast while his exact location will not be transferred to the weather service server, which might be susceptible to an attack that can reveal user locations.

4.1 Solution Design

We present a solution that obfuscates the real location and gathers information locally, so users can learn about the impact that tracking may have. The system acts like an interface between the user's real location and the applications that use it and it is presented in Figure 4.1.

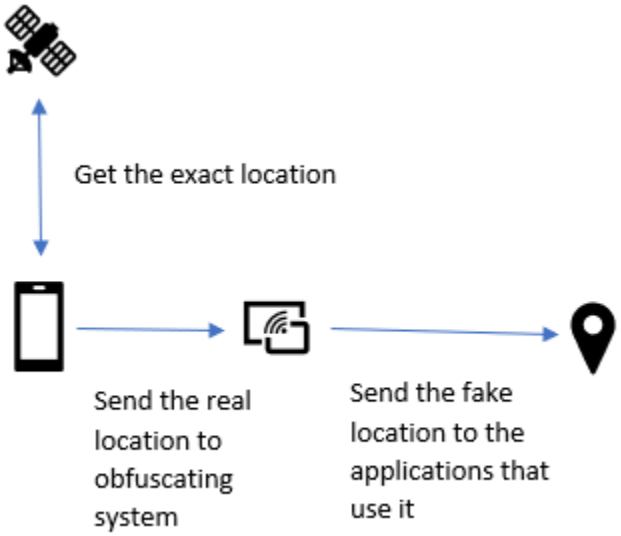

Fig. 4.1 Location obfuscating interface in the context of mobile applications

The main requirement of system is to follow the rules of differential privacy, meaning that the hamming distance between the real location and the fabricated one should not create a quality loss higher than ϵ. The solution retrieves the exact location, applies the obfuscating algorithm and then provides the fabricated coordinates.

We expanded the idea of noise adding, since the basic algorithms turned out to be prone to different types of attacks. Our main idea was to use the dataset of popular locations and we computed the possible fake locations based on it. The map was divided into cells, each one containing a popularity score, ranging from 0 to 9. The required input was a pair of latitude-longitude indicating the location to be obfuscated and the level of obfuscation, ranging between real (no modifications were applied), low (the fabricated location was placed not far from the original location), medium and high (the fabricated location will be placed away from the original location).

Then, the location received as input was placed within a cell by determining the closest distance between the input coordinates and cell coordinates. After finding the cell where the original location belongs, it iterated over the surrounding cells and marked those with a reasonable popularity score. Then it selected one random cell from this collection and gave a pair of latitude-longitude coordinates as an output.

To further enhance privacy, the output location was not simply randomly selected, but the random function is a weighted function. Each cell has a specific weight, proportional with its score. A cell with a higher score has higher, but not absolute, chances of being selected as output.

For location tracking, we added a section that performs the logic of determining where a new fabricated location could be placed, based on the user's historic locations. When a real location is being processed and it is close to a fake location, these two locations help identify the orientation of the user. The distances between consecutive fake locations are proportional to the distances between their corresponding real locations, giving more plausibility to the fake trail.

4.2 Insights

In order to test the tracking algorithm, we created a scenario of a person travelling across Beijing, leaving a trail of five different real locations. These locations served as input for both the last and the current generations of this algorithm. The algorithm treated the array as a series of locations, and each run remembered the output of the last run, making it run in a stateful manner. Figure 4.2 shows all three sets of locations. With yellow markers, we represented the real trace in this scenario. The numbers on each marker show the order in which the markers were generated. The blue markers show the output of the stationary version of the algorithm. According to this result, an attacker would find out that the user has supposedly travelled West from position 1 and then South-East a considerable amount. Depending on the time frame available, it may lead to the user losing credibility in front of the attacker. The red marks the output of the algorithm described in this paper. At a first glimpse, it appears that the fifth location is missing from the result set. In fact, there are five locations but the third and fourth ones are merged. It is immediately visible that the real trace and the "red" trace are very similar regarding the heading, which was the goal of this algorithm. Each one of the "red" locations is placed on a relatively popular cell, therefore the location is plausible. An attacker without prior knowledge about the victim is not able to distinguish this fabricated trace from a real trace.

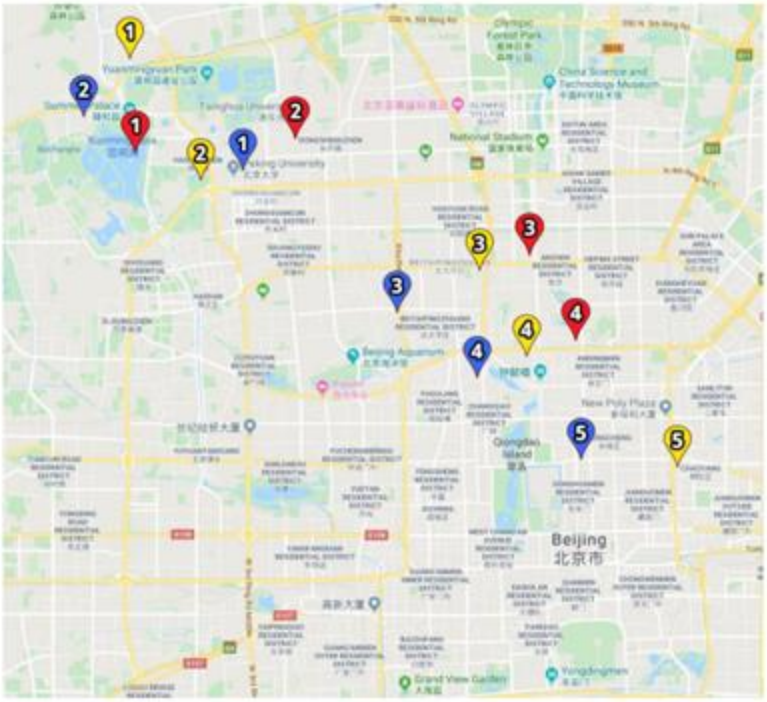

Fig.4.2 Comparison of the current algorithm with the stationary version and the real path

4.2.1 Insights from a security point of view

The first attack that the algorithm was tested against was the same-origin attack. A random input location was selected. This location is stationary and it became the origin of the obfuscation requests. In the next phase, a fabricated location was generated 1000 times, using the same input. Although paper [88] states that less runs are necessary in order to leak a real location, the result became more relevant when more iterations are performed. Figure 4.3 shows the results.

The clusters are scattered across the city, with no relationship between the position of the cluster relative to the origin and the density of the cluster. There are larger clusters in the bottom-left quadrant because that is where some of the most popular areas are. As a general rule, the density of a cluster is only proportional to the popularity of that area of the map. As a precautionary measure, if the real location of a user happens to be in such a popular place, no false location will be generated in that place. The output locations will only be generated in popular places different than the origin. The algorithm is not vulnerable to the same-origin attack, because the distribution of the generated locations on the map does not point towards the original location.

The second type of attack is described as testing the algorithm with multiple different inputs, hoping to identify the input that generates one specific output. In general, the attacks of

this fashion are called brute-force attacks. This type of attack is mainly effective in the case of deterministic algorithms, but to some extent can be used to guess approximate input areas when deployed against location obfuscation algorithms. The attacker starts with a given output (obfuscated) location. For instance, we can assume that this output location is placed at the North end of a large city. The attacker feeds the algorithm with different input locations, attempting to generate various outputs. While he may not be able to choose between multiple apparently valid inputs, he will most likely notice that inputs close to the South end of the city do not generate outputs in the North (because it is too far). Based on this observation, he is able to place the real location of the user somewhere towards the North, or the city center, or slightly towards North-West or North-East. For a given fixed output, we will run the algorithm with various inputs and analyze the results.

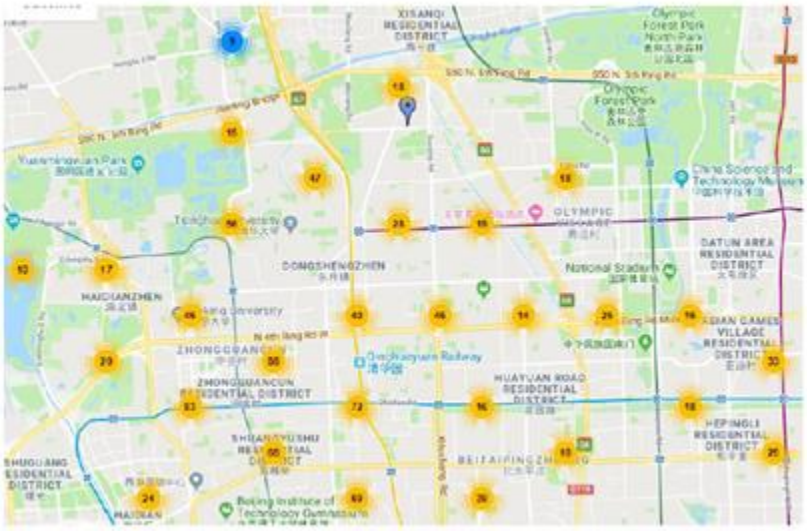

Fig. 4.4 Same-origin attack against our location obfuscation solution

The behavior of the algorithm varies considerably with respect to the position of the designated output. It scans a limited area around the input location and chooses a random map cell that is more popular than a given popularity threshold. Naturally, if a map cell does not have a high enough popularity score, it will never be chosen as output. If the designated output falls in any one of these cells, this attack fails because there will be no input that could generate the expected output. For the edge case where the output location is placed in the only highly popular spot in a region, the algorithm is expected to return that map cell very often.

In a regular scenario, the map is likely to contain multiple popular locations in the fairly wide area surrounding the designated output. A regular scenario was tested and the result was plotted in Fig. 4.5. This bar graph was generated with data collected based on the same map and

input location that were used for the rest of the experiment. The score of the map cell where the input belongs was changed with each iteration, while all the other scores remained the same.

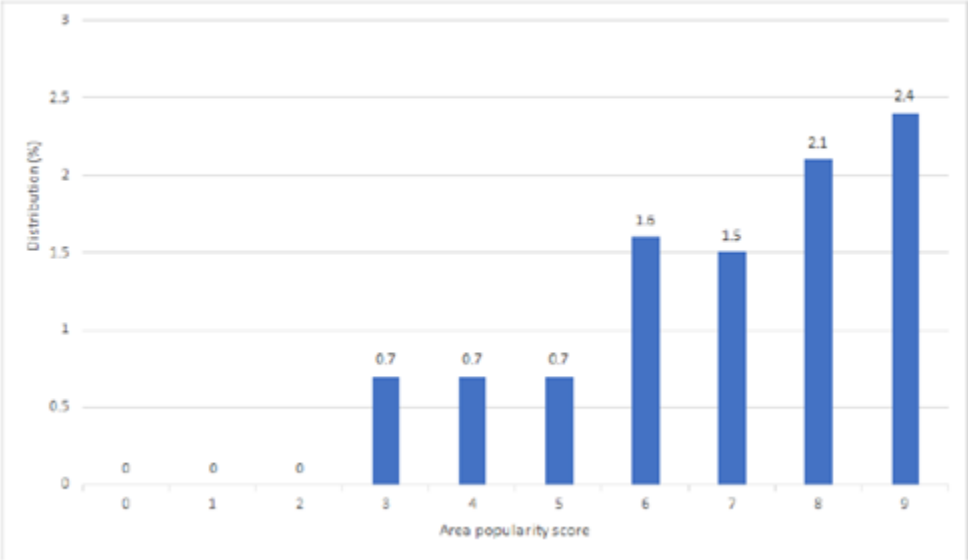

Fig.4.5 Location distribution based on a fixed output

The popularity scores are normalized to fit in the range from 0 to 9, 0 being an area with no interest from pedestrians and 9 being a highly popular area for many citizens. When the popularity score of the input location area is lower than a certain threshold, currently set to 3, there are no locations generated there. Then, as the score increases, some of the generated locations are placed next to the input location. Since the scores map is fairly balanced, even with a maximum score of 9 points, only 2.4% of locations are placed in the observed area. This balance is enforced by the rule that even if an area has the maximum score, it is not guaranteed to be selected. Instead, it only has a slightly higher chance (weight, in a weighted random distribution) compared to other areas with scores of 8 or lower.

5 SEARCHING THROUGH DATA

In this chapter, we present two use cases on privacy over big data. We are interested in searching through encrypted sensitive data, without needing to decrypt it. The first use cases is related to network traffic, especially traffic generated by companies, when a network administrator would be interested in obtaining insights without putting the employees' privacy at risk and the second focuses on experiments with basic financial private data.

5.1 Sharing of Network Flow Data across Organizations

The approaches that we present in the chapter follow two different schemes, presented in Figures 5.1 and 5.2. In both schemes, there is a trusted authority that generates the keys. The information is stored on external servers, which may be 'honest, but curious' and sensitive data must be protected. This creates the need of storing encrypted data, instead of plaintext, to protect privacy.

The first solution, in Figure 5.1, is based on symmetric key encryption, where a trusted party generates the master key that will be used to encrypt the traffic flows and then for match operations to search through data. The master key cannot be sent to users. In this case, the data owner encrypts the data and then it can store it to an external server. When the users need information, the data owner sends queries to the storage server, which runs the matching operations and returns the results, which are then forwarded.

In the second case, in Figure 5.2, the trusted party generates a pair of public and private keys. The users can get the public key used for encryption from the authority and they can send the encrypted output directly to the server. A central trusted entity, that owns the private key, will run the queries and decrypt the matching data.

Considering the analysis of network flows in a company, the first case means that all the plaintext information will be passed through a server that will encrypt everything and forward data to an external storage. In the second case, the logs will be sent encrypted from different points, such as users' computers, but only an administrator will be able to search through them.

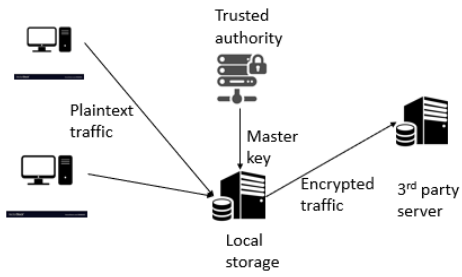

Fig. 5.1. Symmetric-key encryption; the traffic is encrypted on the trusted server level

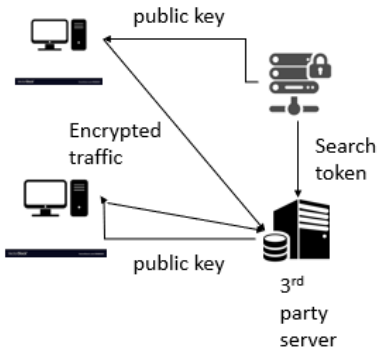

Fig. 5.2 Asymmetric-key encryption; the traffic can be encrypted on the trusted server level

In this chapter, we started from state of the art implementations, which we extended and customize for network flows. The network administrator, as the owner of the data, received a key to search through network flow, that match a certain predicate. Data flows contained IP addresses and ports. We implemented range queries and extend the algorithms to also decrypt the traffic that matches the conditions.

For instance, consider that all networking traffic should be done through SSL/TLS and so going to destination port 443. Any suspicious traffic could be considered having the destination port 80. When a query is issued by the network administrator, he computes the trapdoor and sends the encrypted information to the server. There, it runs the match and decrypt functions and sends the response back. In case of a predicate-only system, it sends a True/False response. Otherwise, it sends back the decrypted IP address or null in case of non-matching flows.

5.1.1 Experiments

For symmetric key implementations, we choose to compare our implementation, based in Inner Product [91] systems with a state-of-the-art solution called HXT [67], and based on

inverted-index. To present the time differences, we choose three queries that return 1%, 3% and 5% of the database.

On the other hand, for asymmetric based solutions, the match and non-match operations take the same running time. In this case, differences are brought by the size of the arrays or the size of the subset. In this case, we use the same database sizes, but without varying the type of queries.

The algorithms were tested on Ubuntu 16.0 server that has 4GB of RAM. The Hidden Vector Encryption[59] and inner-product based algorithm were written completely in C++11, using the PBC [94] and GMP [95] libraries, while the HXT implementation was retrieved from [96]. This uses Scala programming language and Hadoop.

5.1.1.1 Symmetric encryption based experiments

In these experiments, we compared the output of the Inner Product based algorithm, the one that we extended, against the HXT state-of-the-art solution. We ran multiple tests, on three databases, one of 1000 entries, one of 2000 entries and the last one on 4000 entries. We used the same database for both solutions and several IP ranges, from networks to random minimum and maximum addresses.

The first part of both algorithms consists on key setup and encrypting the database. These steps are done only once, at the beginning, then the single operation that runs on the database is the decryption. We ran the experiments, firstly, on a database of 1000 entries. For HXT, the starting algorithm consists of the following steps: initial setup configuration (34s), pairing generation (333ms), generating xtags (173ms), encrypting database (30ms) and generate HXT index (68s) For Inner- Product (IP) based solution, the setup and encrypt phases took 102 seconds. Afterwards, we used a database that contains 2000 entries. The initial setup took 212 seconds for the IP-based solutions. In case of HXT, the xtags, HXT generation and the database encryption take twice as long, proportional to the database size, while the initial setup and the pairing generation remain unaffected by the change. In case of a database with 4000 entries, we measured a setup phase of 431 seconds for the inner product based solution and approximately 300 seconds for the HXT algorithm.

We conducted multiple experiments when trying to extract different IP ranges of the database. We were interested to determine how retrieving a certain percent of the database affects the total query time. For each database size, we decrypted 1% of the total entries, then 3% and 5%. We ran several experiments on each use case, and minimum and maximum decryption times for each experiment are provided in Tables 5.1, 5.2 and 5.3

Decryption size of the total database	1%		3%		5%	
IP based solution (s)	62.39	55.6	62	91.8	125.86	103.59
HXT based solution (s)	18.34	18.38	34	18	34	18

Table 5.1 Experiments results in seconds for IP and HXT based solutions for a database of 1000 entries

Decryption size of the total database	1%		3%		5%	
IP based solution (s)	77.6	24	62.8	83.5	96.6	92.6
HXT based solution (s)	35	36	36	34	70	37

Table 5.2 Experiments results for IP and HXT based solutions for a database of 2000 entries

Decryption size of the total database	1%		3%		5%	
IP based solution (s)	57.6	92.8	70.4	107.5	107.7	123.6
HXT based solution (s)	76	78	86	82	84	82

Table 5.3 Experiments results for IP and HXT based solutions for a database of 4000 entries

The differences in the IP based solutions are determined by the range of IP addresses we look for. In case of searching entire networks, starting from the network up to the broadcast address, the decryption time is especially lower. In the opposite case, in case of random lowest and highest IP addresses of the range, the queries can be up to two times slower. These results are expected, based on the way the algorithm is designed. We observed less noise in case of HXT, where the queries are solved in similar periods of times, regardless the IP ranges.

5.1.1.2 Asymmetric encryption based experiments

In case of Hidden Vector Encryption, we compare the “naïve” implementation, where IP addresses are encoded to binary arrays with the “optimized” one, where the IP address is encoded as the single element of the array. In the first case approach, the speed of operations does not depend on the matching condition, but on the size of the network.

For a class A network, the following operations are done: setup (4.13s) and keygen (0.34s). For a class B network, we have the following results: setup (3.97s) and keygen (0.63s). For a class C network, we have the following results: setup (5.29s) and keygen (1.10s) These are done only once. The encrypt and match operations are run for each entry in the database. For class A, we have the encrypt operation running in 2.48 sec and match operation in 0.46s per each element. For a class B network, we have the following results: encrypt is done in 2.43s and match is done in 0.89s. For a class C network, we have the following results: encrypt is done 2.37s and match in 1.31s.

In the optimized approach, the speed increases since the arrays contain only an element. The global operations done only once, are the following, setup, that took 461ms and token generation, that took 62ms. The operations per entry are the following: encrypt, done in 118ms and query in 79ms. Since we computed the operations per entry on the experimental setup, we simulated the results for larger databases, as presented in Figure 5.3.

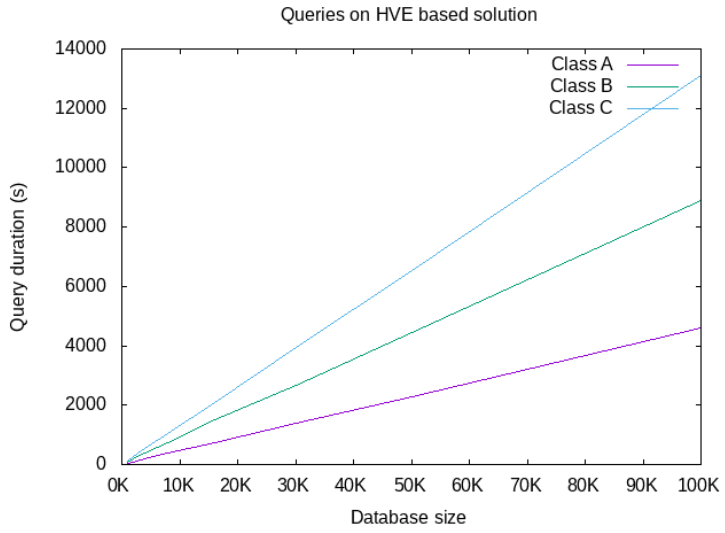

Fig.5.3 Simulations of the optimized HVE solution

An advantage this solution offers is the possibility to run several operations in parallel. There is no data indexing and several steps do not need to be done sequentially. We built a parallel version of the HVE algorithm using the C *pthread* library from POSIX [97] on a four CPUs virtual machine. We parallelized the optimized algorithm, where the vector contains a single element. In this case, the maximum possible number of threads is four and they can be used at setup, where the algorithm uses *3*l+1* random blinding factors, *l* being the length of the vector. For this function, we obtained a 3.55x improvement over the single threaded execution. At encryption, the maximum number of possible threads is three, since there are only three variables that can be computed in parallel. In this case, the improvement is 2.6x over the single threaded implementation. Token generation and query can be parallelized using two threads, in this case of a single element vector, leading to a 1.7x improvement. A comparison between the two serial and parallel implementations is presented in Figure 5.4.

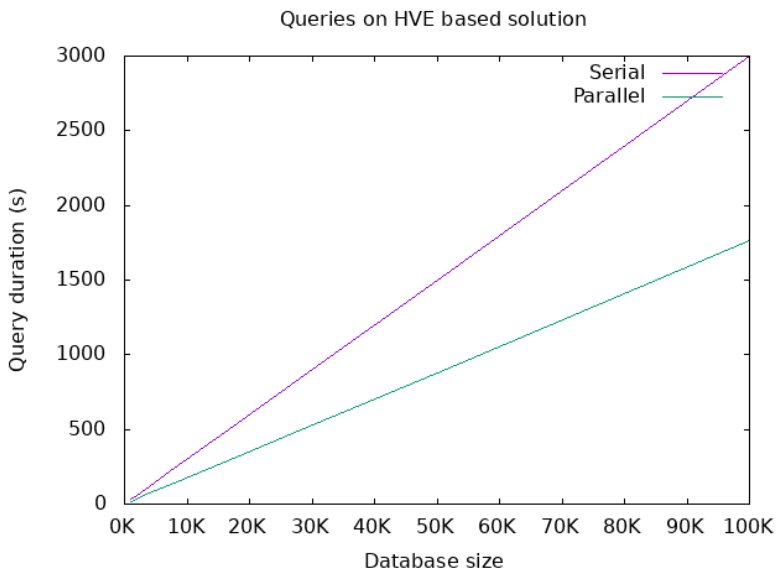

Fig.5.4 Simulations of the HVE solution for serial and parallel implementation

Considering the above experiments, the symmetric key approaches run faster, but in these cases a single user can encrypt and decrypt data. On the other hand, hidden vector encryption permits multiple users to be able to match over the encrypted ciphertexts. HVE implementation offers scalability, since there are a lot of variable initializations and operations that can run in parallel.

Symmetric encryption permits range queries in case of IP addresses together with exact search of a port number, while HVE allows setting a basic taxonomy of ports and looking through them using subset queries. In case of asymmetric encryption, the way data is encoded is important, leading to much better results that the “naïve” approach, where the IP address is transformed in binary arrays.

5.2 Private search in financial data

The trend of storing information on servers in the cloud raises serious security concerns since the increasing number of breaches has resulted in the theft of private data from millions of users. In case of fields like finance, where data privacy is a priority, encryption techniques are analyzed to offer certainty to the customers. A solution for this problem may be searchable encryption, a concept that permits encrypting data locally, moving it to a remote server and then performing operations directly on the ciphertext without decryption.

This project includes the implementation of several state of the art algorithms. Each of the implemented systems is thoroughly tested in order to understand its capabilities and limitations. We combine the encryption systems and the data encodings to perform queries on a database that contains encrypted information about financial transactions. For this project, a transaction is defined by the following information: the sender's IBAN, the receiver's IBAN, the day, month and year of the transaction and the value of the transaction. Various queries will be performed for the latter element.

In Fig. 5.5, there is a diagram depicting a possible usage scenario for performing queries on a database that holds information about financial transactions. Bob wants to make a payment to Alice. He notifies the bank. The bank encrypts the transaction and stores it on a server. Now, Alice wants to check her account and see if she received the payment from Bob. She notifies the bank of her intention and the bank sends a search token based on her request and a decrypting key. Alice sends the token to the storage server and receives back a list of encrypted database records that match the token. She decrypts the records using the decrypting key issued by the bank.

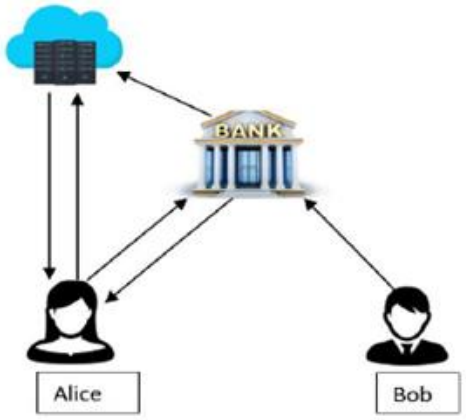

Fig. 5.5 Usage scenario for encrypted database querying

5.3 Insights

In this chapter, we thoroughly analyzed a research-oriented solution, called searchable encryption. Firstly, we built a decrypt searchable encryption scheme, based on a state-of-the art predicate-only solution and we compared it to an "inverted-index" based system. The latter turns to be faster in the majority of testcases, while leaking more information to a possibly "honest-but-curious" server. Our system showed better results in case of ranges that cover a greater percent of the total database, while it spent more time when the match function returned negative results.

Considering the above experiments, we reached the conclusion that the symmetric key approaches run faster, but in these cases a single trustful user can encrypt and decrypt the plaintext data. On the other hand, an asymmetric-key solution, such as hidden vector encryption (HVE) permits multiple users to run queries over the encrypted cipher-texts. HVE implementation offers scalability, since there are a lot of variable initializations and operations that can run in parallel.

Symmetric encryption permits range queries in case of IP addresses together with exact search of a port number, while HVE allows setting a basic taxonomy of ports and looking through them using subset queries. In case of asymmetric encryption, suitable data encoding formats lead to faster algorithms.

We than conducted experiments on financial data and we obtained promising results, including decrypting a small database with 100 records in approximately 0.3 seconds. However, further improvements are needed for searchable encryption techniques to become suitable for real-life applications.

6 PROCESSING BIG DATA

Text processing is one of the most common applications of machine learning today and a suitable example on how big data should be processed. In this chapter, we present ways of classifying text using a custom model or state of the art algorithm and we analyze the overhead that privacy is bringing to the algorithms. The main purpose of this chapter is to understand whether privacy can be implemented in machine learning algorithms, which are the proper conditions and its limitations. In the second part of the chapter, we present a text classification algorithm that runs of short technical texts. Its purpose here is to enhance the need of privacy, proving that insights can be obtained from any kind of data.

6.1 Text classification using Federated Learning

In this section, we used a state of the art algorithm to run text classification and we then added privacy to the scheme. We ran some tests to determine the overhead for texts of different lengths and reached some conclusions from the results obtained.

First of all, we prepared our dataset containing scientific articles regarding solar energy and drugs domains. Then, we started a cleaning process in order to obtain a valid database. We removed non-alphabetic words, as well as the stop words. The final step was lemmatization, used to convert every word to its dictionary form. Them, we transformed every word to a corresponding number.

Regarding the model used, we chose a simple one layer GRU model with sigmoid activation function. We proposed an architecture with two workers, named suggestively Alice and Bob, with their own private data as shown in Figure 6.1. From our database, we sent half of the data to both machines. We had a central node (or a Federated Learning Server) which served as an aggregator for the updated models. Therefore, when our model was sent to both Alice and Bob instances, the model was continuously updated and personalized with their own data, then sent back to the central component, where the complete model was updated correspondingly.

For training and evaluation, we truncated our data to 5000, 2000 and 1000 words per article. On both central and federated approaches we got similar accuracy. The only difference was in time, the distributed model being 1.5x, 3x, respectively 4.5x longer, but with the advantage of privacy. We tested with only two workers, named Bob and Alice, using PySyft framework [71].

Next, we wanted to test our hypothesis that with less data for every worker, the difference between central and federated model would increase in time. So, we used the SMS Spam Collection Data and truncated every tweet to 30 words. The federated model was between 15x and 18x slower than the central one. This may seem a high performance decrease, but we

used only two workers, and every worker had very little data to process. In this case, the computation part was mainly sending and receiving the model updates between the workers and the central node, and because of this the difference in time is so high.

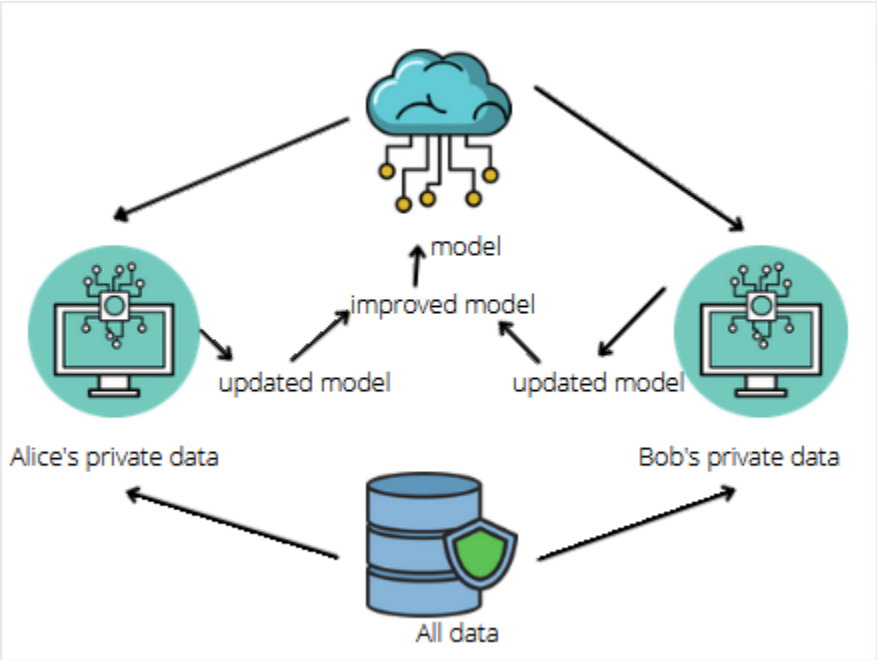

Fig. 6.1 Proposed solution for Federated Learning approach

For the article classification, we truncated the length of an article to 5000, 2000 and 1000 words. We did that because we want a better comparison of the algorithm behavior. After we had got the results, we noticed that the time for the federated approach tends to be larger if the length of the article is smaller. This is because if we have two workers that do very little work during the training (because of the small data), the overhead due to the receiving and sending the data to the central node will be bigger. For 5000 words per article, the distributed approach is 1.5x longer on average, for 2000 words we got 3x longer, for 1000 words 4.5x and for the tweets that have a length of 30 words, the federated approach takes 18x time more.

All the tests were run using Google Colaboratory[1]. We chose this due to its simplicity and good hardware specifications:

- 2vCPU @2.2GHz
- 13GB RAM
- 100GB Free Space

6.1.1.1 Drugs versus Solar Energy

For 5000 words per article, we can see in the Table 6.1 that the federated solution takes less than twice the time that regular PyTorch takes. It is 1.35 time longer, which isn't much given the additional features that we added. For the central model, we ran for 4 epochs, whereas for the Federated approach for only 2, because it was taking too long. As you can see, even if we run for only 2 epochs in contrast to 4 epochs for the central model, we got a pretty close accuracy. Very likely, if we had run for another 2 epochs, the accuracy for the federated model would have been very close to the central one.

For 2000 words per article, we got similar results regarding the model accuracy. We trained for both central and distributed model for 10 epochs. However, we noticed that the overhead in time was increasing, the federated approach lasted four times longer this time. The comparative results are exposed in Table 6.2.

Table 6.1 Comparison between Centralized and Federated results for articles of 5000 words

	Central model	**Federated model**
number of epochs	4	2
accuracy	78.83	72.73
Total time	7h27min	5h24min
Time per epoch	1h52min	2h42min
Privacy?	NO	YES

For 1000 words per article, we get almost identical accuracy. We train for 15 epochs for both central and federated model. As expected, the federated model takes longer to complete, this time lasts 4.5 longer than the normal approach.

Table 6.2 – Comparison between Centralized and Federated results for articles of 2000 words

	Central model	**Federated model**
number of epochs	10	10
accuracy	78.5	78.11
Total time	1h57min	6h
Time per epoch	12min	36min
Privacy?	NO	YES

Table 6.3 – Comparison between Centralized and Federated results for articles of 1000 words

	Central model	Federated model
number of epochs	15	15
accuracy	72.78	72.79
Total time	1h8min	4h31min
Time per epoch	4min39sec	18min
Privacy?	NO	YES

6.1.1.2 Photovoltaic versus Renewable energy

In this section, we get the results for two more similar domains: photovoltaic and renewable energy. As we expected, we got worse results than the previous run where we had more unrelated topics. We tested in the same manner, for a length equal to 5000, 2000 and 1000 words per article.

For the 5000 words test, we got an accuracy of about 65%. Like in the previous run, the federated approach takes 1.5x more. For the 2000 length, we ran for 10 epochs. The accuracy is approximately 61% and the federated model takes 2.5x more time than the central one, similar with what we got in the first section. For 1000 words length, as we have expected the federated approach takes much more: approximately 4.5x more, which checks our expectations.

The results are exposed in Tables 6.4, 6.5 and 6.6.

Table 6.4 – Comparison between Centralized and Federated results for articles of 5000 words

	Central model	Federated model
number of epochs	4	2
accuracy	59.9	64.59
Total time	6h12min	4h36min
Time per epoch	1h33min	2h18min
Privacy?	NO	YES

Table 6.5 – Comparison between Centralized and Federated results for articles of 2000 words

	Central model	Federated model
number of epochs	10	10
accuracy	60.69	57.85
Total time	3h16min	7h51min
Time per epoch	19min	47min
Privacy?	NO	YES

Table 6.6 – Comparison between Centralized and Federated results for articles of 1000 words

	Central model	Federated model
number of epochs	15	15
accuracy	61.71	57.5
Total time	1h13min	4h14min
Time per epoch	4min48sec	17min
Privacy?	NO	YES

6.1.1.3 SMS Spam Collection Dataset

In this section, we tested our model on the SMS Spam Collection Data Set2. We truncated the length of every tweets to 30 words to be able to train for 100 epochs. The accuracy is high, as expected, because of the number of epochs we trained. What is interesting is the time difference between the two approaches. As we noticed so far, less words per article, more time the federated model takes, which is confirmed here as well. For 30-word tweets, the distributed model takes no less than 12x more in time than the central one. This behaviour can be justified by two reasons: the number of workers is very low (we used only two, Bob and Alice), and, the second one would be due to the fact that using few workers with less data, the most time consuming part would be sending and receiving the data between the workers and the federated learning server or the central node. The results for 100 epochs can be seen in the Table 6.7 down below.

Table 6.7 Comparison between Centralized and Federated results for short texts

	Central model	Federated model
number of epochs	100	100
accuracy	98.04	96.17
Total time	8min	1h31min
Time per epoch	4.8s	55s
Privacy?	NO	YES

One insight obtained from the results is that the smaller the input are , the greater the time difference between the central and the federated methods is. This statement is valid for constants number of workers, of course. The time difference can be justified by the fact that using small data, most of the time will be lost by sending and receiving the updated model between workers and the central node. If we increase the number of workers, this aspect will not affect the training time for a particular device, because every worker has its own data, but the accuracy of the model would be improved. So, more workers/devices mean better predictions and more personalized models. In a real life scenario, there will be billions of devices (such as smartphones or smart watches), so because of the tremendous amount of data, a very good accuracy is guaranteed.

The communication between the central server and the workers is a classical bottleneck in Federated Learning. This issue affects the maximum capacity of a model that can be sent and its accuracy. The mitigation of these problems could be to reduce the model architecture or to remove some parameters. This will cause a worse accuracy, but the user experience will be improved. Another approach would be to train some subsets of the global model on every device, a method named Federated Dropout, further detailed in [99].

6.2 Model based classifier for short, technical texts

In this chapter, we analyze text processing in cases where machine learning algorithm fail to return optimal results.

The starting point of the project is a specialized database, containing information, datasets and articles on renewable energy and solar data. The main issue related to these types of databases is the high number of unstructured information within. Any search will raise a high number of results, which may be confusing to a user. A lot of essential information can be lost after the first pages and many connected datasets may be lost due to a high number of retrieved results. Despite the knowledge that a complex database provides, the difficulty of retrieving specific data may overcome the benefits.

The purpose of the implementation is to build a classification algorithm for short texts, including datasets, which contain some prominent technical words and are hard to be classified by the state-of-the-art solutions.

The main scope of the project is to build a “smart” search system, which is able to associate scientific papers with web services, to get data from the multiple sources in a coherent way and to show the results based on their relevance. For users, this process is transparent, they start by searching in the webserver using keywords or free text and, in the backend, there is a classifier that provides not only the results obtained by querying, but also entries related to the subject, which may be useful for an expert. Figure 6.2 explains the big picture of the process.

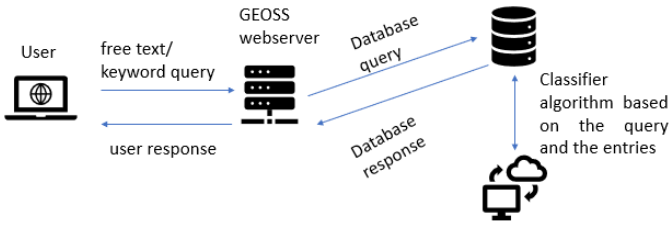

Fig.6.2 Classification algorithm integrated in the existing system

In case of short texts, such as metadata records, the best approach is to build up a hierarchy of predefined words related to the topic and automatically assign each text to these categories. In this case, an expert view is needed to provide the necessary vocabulary and the hierarchy. In this case the tree was built from general to more specific information related to energy, starting by the most general level, there is "Theme" →"Domain" → "Sub-domain" → "Information" → "Measures".

- At the "Theme" level, we have "Energy".
- At the "Domain" level, there is "Renewable Energy".
- At the "Sub-domain" level, there is "Solar Energy".
- At the "Information" level, there are "Solar resource", "Atmosphere and meteorology", "Ground topography", "Meteorological Year" and "Solar potential", together with the most prominent terms of each category.
- Each "Measures" entry is linked to each category above and provides further information on the technical vocabulary for each case. Words such as "time sampling", "time resolution", "numerical weather prediction model" are considered relevant for this level.

6.2.1 Observations

After the classification algorithm was run, the information was divided into five fields, on the main tracks of the subjects. Every text was attached with a similarity score to a subject. The first comparisons that we did are against manual searches.

The first technical field is 'solar energy' and the top ten matches are related to global solar exposure, solar atlas on solar potential across countries developed by research institutes, The Bureau of Meteorology's (BOM) [111] computer radiation model on solar exposure and different irradiation maps divided by region. In comparison, the manual search of the same concept retrieves results connected to ground-based measurements for stations in the Renewable Resource Monitoring and Mapping (RRMM) Solar Radiation Monitoring Network. These types of content are similar and the results can be automatically clustered into categories. Our solution provides 285 accurate results, while the manual search 790.

The second technical field is 'Atmosphere and meteorology'. In this case, the best matches are related to datasets on wind intensity, geothermal temperature and meteorological stations. Also, some results that do not contain the keywords in title, but provide interesting input in the abstract, were found. For example, the algorithm mapped ground-based measurements for stations in the Renewable Resource Monitoring and Mapping (RRMM) Solar Radiation

Monitoring Network. These provide one-minute resolution data for three solar components, along with temperature, humidity, wind speed and direction, and barometric pressure. Even the title of these resources is rather related to solar components, the content provides also insightful data on meteorology. Here, a keyword search will provide significantly less results. The algorithm provides 165 exact matches on the subject. In case of manual search, we find only one result if we search for both terms at the same time.

The third main topic is 'ground topography'. The algorithm obtains 165 matching results, while the search query provides no results in the database. Searching separate terms leads to 510 results. The top automatic classification results contain datasets about railroad and road linestrings, airport points, transmission lines, protected areas, country administrative boundaries and elevation multipolygons. These words match the predefined expert's labels. The manual search for this topic considers wind speed maps as the most relevant results.

The fourth field is 'meteorological year'. The algorithm returns as most relevant results the hourly, daily, monthly and yearly irradiation maps, averages of daily global solar irradiation, monthly average solar resource and annual wind maps. The manual search return as best results similar entries, datasets containing the annual average Solar Radiation (DNI, GHI, BHI, DHI), yearly irradiation maps, but also daily values of Surface Solar Irradiation, which may be less connected to the subject. An interesting result that we easily found, but considered less relevant by the manual search, is a wind atlas, meaning annual mean wind speed and specific production maps at four levels (25, 50, 75 and 100 m) above ground and sea. We also found a global map of 22-year monthly & annual average atmosphere pressure provided by NASA, while the manual search retrieved maps life loss expectancy for different sectors of activity.

The fifth field is 'solar potential'. The top results are similar for the automatic and manual search, related to technical and theoretical potential for solar energy production. The next results provided by the manual search are related to solar irradiance, as the ones in the previous labels, while the classification system finds datasets of photovoltaic electricity output and concentrating solar power (CSP) project opportunity areas.

6.2.2 Insights

Analyzing the results above, the best approach related to clustering metadata records of solar observations is a hybrid approach including a mix of manual labels and machine learning approaches. In this case, a list of predefined labels is necessary, since the themes, domains or requested information are difficult to be discovered automatically. The main challenge of classifying the entries is related to the length of the texts and the technical concepts. It is highly important to know beforehand what is relevant from an expert's perspective and the machine learning algorithm can provide a new insight over text similarities and interesting results. We can get metadata records that are not automatically returned by the database when querying. We

can use a ranking system, based on the search text, so the most relevant results will be shown first, for a better user experience. Thanks to NLP and ML algorithm, we can connect in the future resources from multiple sources, so gathering information in a field will be easily obtained from a single entrypoint.

7 CONCLUSIONS

The work in the thesis explores several solutions for privacy at each level of data processing. Since big data and privacy are complex and general terms that lead to wide research opportunities, it is unfeasible to build generally secure layers, covering multiple scenarios and types of data.

The present thesis focuses on improving privacy for big data, while following a list of re-defined processing steps: gathering, collecting and transmitting, storing and processing data. The work in this thesis covers multiple fields, from security and privacy to natural language processing. It contains original contributions to the field of privacy-preserving cryptography, location obfuscation, security in IoT, open education and natural language processing, enabling further research on any of these areas.

All the contributions have been published in [9], [10], [11], [12], [13], [14], [15] and [16]. Working on the thesis required different technical skills, from using multiple programming languages to understating of algebra structures such as groups, rings and fields. Interconnecting all these fields contributes to the state of the art, thus enabling the development of more effective privacy-oriented solutions.

All the work can be divided as following:

Contribution 1: Main protocols in IoT and their security solutions

Articles:

- *Survey of standardized protocols for the Internet of Things* [9]
- *Challenges in security in Internet of Things* [10]

This contribution is rather an exploration into the networking stack of embedded devices, the main protocols and the adapted security solutions. Generally, the IoT devices have low memory, reduced battery capacity, reduced processing capabilities and vulnerable radio conditions. The standard TCP/IP stack is not suitable for this environment, so working groups have started to adapt the existing protocols to new versions. An addressing scheme, such as IPv6 was taken into consideration, since there are billions of interconnected nodes. Multiple working groups already started to standardize IoT specific protocols, such as 6LoWPAN (RFC 4944 and RFC 6282), IEEE802.15.4 and ZigBee and to describe ways of enabling IPv6 in constrained environments. Other found requirements refer to security and privacy, since the number of Denial of Service attacks has recently increased. At the application layer, a common way of retrieving and requesting data is using Web architecture, and more specific, HTTP. This uses URIs as resource identifiers and it is based on REST architecture to publish information. For embedded devices, there is a Constrained RESTful Environments (CoRE) IETF working group, that aims to develop

RESTful protocols, compatible with HTTP for resource-constrained devices. The most prominent protocol of this kind is CoAP, at the application layer. We presented a survey of the most used standardized protocols for Internet of Things. We investigate application layer protocols (CoAP, MQTT), service discovery protocols (mDNS, DNS-SD, uBonjour) and infrastructure protocols (IEEE 802.15.4, 6LoWPAN, LoRaWAN). We presented different ways to integrate application-layer protocols such as MQTT, CoAP and HTTP. Finally, we propose a smart home system with wireless sensor nodes and robot assistants that use standardized IoT protocols (IEEE 802.15.4, 6LoWPAN, CoAP).

We extended the work from the first article with a security overview of these protocols. We reached the conclusion that standardization of the networking stack and security is a key factor for the success of IoT. A networking stack for IoT systems can be created using, at each level, protocols developed especially for constrained environments. At the physical and MAC level, there are 802.15.4 based protocols, which can be found on multiple devices. At networking layer we can use a protocol that provides routing based on MAC addresses. A new adaptation layer is added at the stack between network and transport, used for providing IPv6 addresses to IoT devices. At the application layer, the protocols were adapted for carrying less data more often. Learning how to develop a smart system for buildings, campuses and homes leads to gaining knowledge on both the advantages and issues related to connection multiple and heterogeneous devices. On one hand, they can improve the quality of life and resource usage, but they also come with security and privacy risks. Adding security to IoT is a challenging task, since the most secure mechanisms in the TCP/IP stack are not suitable. The standard DTLS protocol can be used, but together with improvements that lead to less overhead. Also, new security protocols can be developed, when trying to minimize the initial handshake.

We proposed a security solution for the above networking stack based on pre-shared keys, together with an existing algorithm. We decided to use key derivation with AES-128, since it is hardware supported on many IoT devices and to add new options “AUTH” and “AUTH_MSG_TYPE” for messages between server and client. Every client has a key configured and the server has a pool for each id. The first message sent from the client to the server contains the client id and a unique token. The server finds the password associated with the device, derives the key, nonce_1 and sends an encrypted message as a challenge. When the client gets the challenge, it uses the shared key to decrypt the packet, gets the key and nonce_1 and sends a response using the derived key and the token. If the token is the same as in the initial transmission, then the client is authenticated. We encrypt the application layer payload and use the options in CoAP header to separate encrypted from plain-text queries.

The main issue is that pre-shared keys can be broken, so our system proposal looks for possible attacks, by analyzing traffic patterns at the server side. If malicious devices were

authenticated, the users must be alerted and the system should be configured with another solution, such as asymmetric keys.

Contribution 2: Geo-indistinguishability for mobile users

Articles:

- *Teaching privacy through the development and testing of a location obfuscation solution* [11]
- *Location privacy for non-stationary users* [12]

The need to maintain location privacy is only stronger in today's environment, when context-awareness, location-awareness services are everywhere. Current systems are able to track anyone in real time, with incredible accuracy. Together with technological advancements in processing power and capacity, the population can be followed wherever they go, much easier than ever before. For instance, one's location can be used maliciously to send spam messages containing products or services related to the victim's location. Another risk of ignoring location privacy is given by physical abuse. An attacker might use someone's location to follow around, rob, assault this victim. A third reason why the lack of location privacy can cause harm to a person is that the location can tell things about a person. Just by knowing where someone was in the past period of time can lead to fair assumptions about religion, political orientation, health status and other similar private information.

We started from a basic obfuscation algorithm, that uses the Planar Laplace distribution to add noise to a real location and we proved that it is not reliable enough. From a security standpoint, triangulation is a threat to locations obfuscated with the basic algorithm. Additionally, some locations generated that way may not be plausible, because they can be placed in either odd or normal areas with the same probability. Both these flaws and any others related to them have been addressed by developing a new version of the algorithm. The results, presented as a comparison between our solution and the state-of-the-art starting point, highlight a significant improvement. Every location generated with our algorithm belongs to a relatively popular area. The only criterion for generating locations is the popularity score. The distance between the real and the fabricated location is no longer relevant to the algorithm. The maximum distance is limited by the obfuscation level specified as input, but it is not enforced in any other way.

We developed the algorithm to create credible path for mobile users. We generated new locations based on the popularity score of the points of interest nearby the user and on his/her pervious location. The results are consistent with the theoretical expectations. Given a trajectory based on real locations, the output appears very similar in orientation, with locations generated in plausible places.

After building the algorithm, we tested it against state-of-the art attacks over location privacy obfuscators. The algorithm behaves similarly in most cases, but there are a few edge scenarios to be taken into account. When the obfuscated location, somehow, is placed in an unpopular place on the map, the algorithm will never be broken by the brute- force attack. When the area of the obfuscated location is minimally popular and some more popular areas are in the vicinity, the algorithm is likely to choose those popular places as output. This may lead the attacker into believing that the area containing the real location is smaller than it actually is. However, the usual scenario matches the behavior of the original algorithm, meaning that our solution is also offering protection against the brute-force attack.

Contribution 3: Secure data storage

Articles:

- *Practical analysis of searchable encryption strategies for financial architecture* [13]
- *Sharing of Network Flow Data across Organizations using Searchable Encryption* [14]

Analyzing traffic patterns can help organizations provision better in terms of employee time assignment and equipment purchases and can also assist in improving inter-organizational collaborations. Therefore, sharing network flow data can significantly improve the optimizations of operations. As a related aspect, some inter-organizations tasks, e.g., cybersecurity incident response, require multiple organizations to jointly investigate their network logs to understand the nature and the scope of the threat. On the other hand, when dealing with network logs, security and privacy become important concerns. Disclosing an organization's network traffic logs may allow an adversary to learn details about confidential business interests. To avoid such privacy leaks, we considered searchable encryption, a powerful tool that can be used to protect the data. Instead of sharing its data in plaintext, each organization first encrypts its dataset with a special kind of encryption that allows search directly on ciphertexts. Then, based on specific scenarios, an organization can grant another party the ability to search and retrieve some of its data based on cryptographic objects called search tokens. Using a token, one can perform search over encrypted data, without the need for decryption. The search is performed based on some predicate. For instance, access can be granted only for packets originating from a certain IP address or to a certain destination port. As a result of the search, only plaintexts that matches the search predicate are obtained. Non-matching data items remain encrypted, and no information about those is revealed during the search. We implemented conjunctive queries, such as diverse combinations of destination IP and port, e.g., HTTP traffic to a server or SSH traffic to an entry point.

We investigated the use of several prominent techniques for search on encrypted data to the specific problem of querying encrypted network logs across organizations. We focused on

network data at flow granularity, which contains metadata attributes that are relevant in most use case scenarios (e.g., IP addresses, port numbers, number of bytes and packets transferred per connection, protocol type, etc). The main challenge arising when using searchable encryption is the computational overhead it brings. When applying state-of-the art schemes directly on network traffic, querying the data takes a lot of time. We found some proper encodings and proposed a customized approach for network traffic at flow granularity in order to bring the computational overhead to practical levels.

We discovered that the symmetric key approaches run faster, but in these cases a single user can encrypt on decrypt data. On the other hand, an asymmetric key approach, such as hidden vector encryption (HVE) permits multiple users to be able to match over the encrypted ciphertexts. HVE implementation offers scalability, since there are a lot of variable initializations and operations that can run in parallel. Symmetric encryption permits range queries in case of IP addresses together with exact search of a port number, while HVE allows setting a basic taxonomy of ports and looking through them using subset queries. In case of asymmetric encryption, the way data is encoded is important, leading to much better results that the "naïve" approach, where the IP address is transformed in binary arrays.

Afterwards, we have implemented two Hidden Vector Encryption schemes and two inner product supporting systems. We have performed a thorough evaluation of their correctness and of their performances for various use case scenarios and for different sets of parameters, for a database containing information about financial transactions. We have chosen multiple encodings for the transaction value that support range queries or exact matches or both. Promising results were achieved, including decrypting a database with 100 records in approximately 0.3 seconds.

Contribution 4: Secure data processing

Articles:

- *Privacy in machine learning algorithms* [15]
- *Short text classification* [16]

Connecting data sometimes imply finding links between research papers or reports which describe the methods, on one hand and the implementation and the results of the software algorithms, on the other hand. Current search engines do not provide links or structured relationships between these components of a document, thus reducing their fitness for use. A knowledge hub aims at providing linkages between these components. The main issue related to these types of databases is the high number of unstructured information within. Any search will raise a high number of results, which may be confusing to a user. A lot of essential information

can be lost after the first pages and many connected datasets may be lost due to a high number of retrieved results. Despite the knowledge that a complex database provides, the difficulty of retrieving specific data may overcome the benefits.

We built a classification algorithm for short texts, including datasets, which contain some prominent technical words and are hard to be classified by the state-of-the-art unsupervised machine learning algorithms. We have compared our solution with manual search and we have found some positive results. We have some matches that do not contain the same keywords as searched, but synonyms and we found relevant results that have been ignored by the manual search engine. We divied the entire database into five categories and our goal was to connect each entry to one of the categories. We found out that the entries related to 'solar energy' are easily classified, since most of the entries are connected to solar observations. We most difficult topic to be examined is solar potential. It contains more general terms, such as 'potential', that leads to more false positive results.

After proving that we can get insights from any type of data, we focused on adding privacy on machine learning algorithms. We started from an existing technology, Federated Learning, that ensures privacy and we ran multiple text classification experiments to see its accuracy and overhead compared to a centralized approach. We ran the tests with two private workers for two completely different topics and for two similar topics and we took into consideration texts of different lengths, varying from 5000 to 1000 words. We found similar results for both the private and the centralized approaches and we remarked a higher overhead in case of shorter texts . Then, we ran the algorithm on a SMS spam dataset, where we found an increase of overhead for Federated Learning. We examined the results and found that the main overhead is caused by information sending between workers and the central server. We reached the conclusion that this aspect is different in the real world. With billions of connected devices, that can server as workers, this limitation can be mitigated.

8 BIBLIOGRAPHY

[1] [Online]. Available: https://www.wired.com/story/the-man-who-saw-the-dangers-of-cambridge-analytica/. [Accessed November 201].

[2] N. Gruschka, V. Mavroeidis, K. Vishi and M. Jensen, "Privacy issues and data protection in big data: a case study analysis under GDPR," in *IEEE International Conference on Big Data, pp.5027-5033*, 2018.

[3] [Online]. Available: https://www.embroker.com/blog/cyber-attack-statistics/. [Accessed November 2021].

[4] [Online]. Available: https://www.securitymagazine.com/articles/95723-data-privacy-in-the-era-of-covid-19-vaccine-rollouts. [Accessed November 2021].

[5] C. Dwork, "Differential privacy: A survey of results," in *International conference on theory and applications of models of computation*, Springer, Berlin, Heidelberg, 2008.

[6] [Online]. Available: https://analyticsindiamag.com/google-comes-up-with-a-new-differentially-private-clustering-algorithm/. [Accessed November 2021].

[7] [Online]. Available: https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf. [Accessed November 2021].

[8] [Online]. Available: https://payu.in/blog/the-big-6-steps-of-big-data-explained/. [Accessed November 2021].

[9] I. M. Florea, R. Rughinis, L. Ruse and D. Dragomir, "Survey of standardized protocols for the Internet of Things," in *21st International Conference on Control Systems and Computer Science (CSCS)*, Bucharest, 2017.

[10] I. M. Florea, L. C. Ruse and R. Rughinis, "Challenges in security in Internet of Things," in *16th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Bucharest, 2017.

[11] I. M. Florea, D. Vornicu, J. A. Vaduva and R. Rughinis, "Teaching privacy through the development and testing of a location obfuscation solution," in *The International Scientific Conference eLearning and Software for Education*, Bucharest, 2020.

[12] I. M. Florea, D. Vornicu, S. D. Ciocirlan and R. Rughinis, "Location privacy for non-stationary users," *University Politehnica of Bucharest Scientific Bulletin,* vol. 83, no. 4, 2021.

[13] I. M. Florea, S. D. Ciocirlan and I. Dura, "Practical analysis of searchable encryption strategies for financial architecture," in *9th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Bucharest, 202.

[14] I. M. Florea, G. Ghinita and R. Rughinis, "Sharing of Network Flow Data across Organizations using Searchable Encryption," in *23rd International Conference on Control Systems and Computer Science (CSCS)*, Bucharest, 2021.

[15] I. M. Florea, M. Constantin and S. D. Ciocirlan, "Privacy in machine learning algorithms," in *20th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Bucharest, 2021.

[16] I. M. Florea, R. Rughinis and S. D. Ciocirlan, "Short text classification," in *20th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Bucharest, 2021.

[17] [Online]. Available: https://hadoop.apache.org/. [Accessed February 2022].

[18] [Online]. Available: https://auth0.com/blog/the-7-most-common-types-of-cybersecurity-attacks-in-2021/. [Accessed November 2021].

[19] [Online]. Available: https://www.the-next-tech.com/blockchain-technology/how-much-data-is-produced-every-day-2019/. [Accessed November 2021].

[20] [Online]. Available: https://www.computerweekly.com/news/252492564/Belgian-security-researcher-hacks-Tesla-with-Raspberry-Pi. [Accessed November 2021].

[21] J. Granjal, E. Monteiro and J. Sa SIlva, "Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues," *IEEE Communications Surveys & Tutorials,* vol. 17, no. 3, pp. 1294-1312, 2015.

[22] K. N. Montenegro G., J. Hui and C. D., "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," RFC 4944, 2017.

[23] S. M. Sajjad and M. Yousaf, "Security analysis of IEEE 802.15. 4 MAC in the context of Internet of Things (IoT)," in *Conference on Information Assurance and Cyber Security (CIACS)*, 2014.

[24] P. M., R. J., M. P. and L. S., "Performance study of IEEE 802.15.4 using measurements and simulations," in *IEEE Wireless Communications and Networking Conference*, 2006.

[25] K. N., G. Montenegro and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals," in *RFC 4919*, 2007.

[26] P. THUBERT and J. HUI, "RFC 6282, Compression Format for IPv6 Datagrams over IEEE 802.15. 4-Based Networks," Internet Engineering Task Force, Fremont, CA, USA.

[27] ALLIANCE, LoRa, " LoRaWAN What is it?-A Technical Overview of LoRa and LoRaWAN," 2015.

[28] A. J. Jara, M.-J. Pedro and A. Skarmeta, "Light-weight multicast DNS and DNS-SD (lmDNS-SD): IPv6-based resource and service discovery for the web of things," in *Sixth international conference on innovative mobile and internet services in ubiquitous computing*, 2012.

[29] S. Cheshire and M. Krochmal, "Multicast DNS," RFC 6762, 2014.

[30] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhar and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE communications surveys & tutorials,* vol. 17, no. 4, pp. 2347-2376, 2014.

[31] S. Cheshire and M. Krochmal, "DNS-based service discovery," RFC 6763, 2013.

[32] R. Klauck and M. Kirsche, "Bonjour contiki: A case study of a DNS-based discovery service for the internet of things," in *International Conference on Ad-Hoc Networks and Wireless*, Springer, Berlin, Heidelberg, 2012.

[33] I. Ishaq, D. Carels, G. Teklemariam, J. Hoebeke, F. Van den Abeele, E. De Poorter, I. Moerman and P. Demeester, "IETF Standardization in the Field of the Internet of Things (IoT): A Survey," *Jurnal of Sensor and Actuator Networks,* vol. 2, no. 2, pp. 235-287, 2013.

[34] Z. Shelby, K. Hartke and C. Bormann, "The constrained application protocol (CoAP)," 2014.

[35] C. Bormann, A. P. Castellani and Z. Shelby, "Coap: An application protocol for billions of tiny internet nodes," *IEEE Internet Computing,* vol. 16, no. 2, pp. 62-67, 2012.

[36] E. Rescorla and M. Nagendra, "Datagram transport layer security," 2016.

[37] D. McGrew and D. Bailey, " Aes-ccm cipher suites for transport layer security (tls)," *RFC 6655,* pp. 1-6, 2016.

[38] "SECG-Elliptic Curve Cryptography-SEC 1," [Online]. Available: http://www.secg.org. [Accessed 17 February 2017].

[39] S. L. Keoh, O. Garcia Morchon, K. S. Sandeep and E. Dijk, "DTLS-based Multicast Security for Low-Power and Lossy Networks (LLNs)," draft-keoh-tls-multicast-security-00, 2014.

[40] S. L. Keoh, S. K. Sandeep and H. Tschofenig, "Securing the internet of things: A standardization perspective," *IEEE Internet of things Journal,* vol. 1, no. 3, pp. 265-275, 2014.

[41] K. Hartke, "Practical Issues with Datagram Transport Layer Security in Constrained Environments," draft-hartke-dice-practical-issues-00, 2013.

[42] S. Santesson and T. Hannes, "Transport layer security (TLS) cached information extension," draft-ietf-tls-cached-info-23.txt, 2014.

[43] P. Wouters, H. Tschofenig, J. Gilmore, S. Weiler and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)," draft-ietf-tls-oob-pubkey-11, 2014.

[44] S. Keoh, S. Kumar and Z. Shelby, "Profiling of DTLS for CoAP-Based IoT Applications," draft-keoh-dice-dtls-profile-iot-00, 2013.

[45] D. Locke, "Mq telemetry transport (mqtt) v3. 1 protocol specification," *IBM developerWorks Technical Library,* vol. 15, 2010.

[46] U. Hunkeler, H. L. Truong and A. Stanford-Clark, "MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks," in *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*, 2008.

[47] A. Stanford-Clark and H. L. Truong, "Mqtt for sensor networks (mqtt-sn) protocol specification," *International business machines (IBM) Corporation version,* vol. 1, no. 2, pp. 1-28, 2013.

[48] D. Thangavel, X. Ma, A. Valera, H.-X. Tan and C. Keng-Yan Tan, "Performance evaluation of MQTT and CoAP via a common middleware," in *IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP)* , 2014.

[49] [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/426248/Acquisition_and_Disclosure_of_Communications_Data_Code_of_Practice_March_2015.pdf. [Accessed February 2022].

[50] M. Andres, N. Bordenabe, C. Konstantinos and C. Palamidessi, "Geo-Indistinguishability: Differential Privacy for Location-Based Systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013.

[51] N. Bordenabe, C. Konstantinos and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014.

[52] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, no. 5, pp. 557-570, 2002.

[53] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg and D. Boneh, "Location Privacy via Private Proximity Testing," *NDSS,* vol. 11, 2011.

[54] R. Shokri, C. Troncoso, C. Diaz, J. Freudiger and J.-P. Hubaux, "Unraveling an Old Cloak: k-anonymity for Location Privacy," in *Proceedings of the 9th annual ACM workshop on Privacy in the electronic society*, 2010.

[55] A.-M. Olteanu, K. Huguenin, R. Shokri, M. Humbert and J.-P. Hubaux, "Quantifying Interdependent Privacy Risks with Location," *IEEE Transactions on Mobile Computing,* vol. 16.3, pp. 829-842, 2017.

[56] [Online]. Available: https://www.wired.com/2007/12/why-anonymous-data-sometimes-isnt/. [Accessed November 2021].

[57] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious RAMs," *Journal of the ACM,* vol. 43, no. 3, pp. 431-473, 1996.

[58] C. Bosch, P. Hartel, W. Jonker and A. Peter, "A Survey of Provably Secure Searchable Encryption," *ACM Computing Surveys,* vol. 47, no. 2, pp. 1-51, 2015.

[59] D. Boneh and B. Waters, "Conjunctive, Subset, and Range Queries on Encrypted Data," in *4th Theory of Cryptography Conference*, Amsterdam, 2007.

[60] D. Boneh, E.-J. Goh and K. Nissim, "Evaluating 2-DNF Formulas on Ciphertexts," in *Proceedings of Theory of Cryptography Conference*, 2005.

[61] V. Iovino and G. Persiano, "Hidden-vector encryption with groups of prime order," in *International Conference on Pairing-Based Cryptography*, Berlin, 2008.

[62] J. Katz, A. Sahai and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Annual international conference on the theory and applications of cryptographic techniques*, Berlin, 2008.

[63] S. Kim, L. Kewi, A. Mandal, H. Montgomery, A. Roy and D. J. Wu, "Function-Hiding Inner Product Encryption Is Practical," in *International Conference on Security and Cryptography for Networks*, 2018.

[64] R. Bost, B. Minaud and O. Ohrimenko, "Forward and Backward Private Searchable Encryption from Constrained Cryptographic Primitives," in *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017.

[65] C. David, S. Jarecki, J. Charanjit, K. Hugo, R. Marcel-Cătălin and M. Steiner, "Highly-Scalable Searchable Symmetric Encryption with Support for Boolean Queries," in *CRYPTO 2013: Advances in Cryptology*, Berlin, 2013.

[66] S. Lai, S. Patranabis, A. Sakzad, J. K. Liu, D. Mukhopadhyay, R. Steinfeld, S.-F. Sun, D. Liu and C. Zuo, "Result Pattern Hiding Searchable Encryption for Conjunctive Queries," in *CCS '18: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[67] S. Lai, S. Patranabis, A. Sakzad, J. K. Liu, D. Mukhopadhyay, R. Steinfeld, S.-F. Sun, D. Liu and C. Zuo, "Result Pattern Hiding Searchable Encryption for Conjunctive Queries," in *ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*, Toronto, On, Canada, 2018.

[68] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM,* vol. 13, no. 7, pp. 422-426, 1970.

[69] F. Sky, S. Jarecki, H. Krawczyk, N. Quan, R. Marcel and M. Steiner, "Rich Queries on Encrypted Data: Beyond Exact Matches," in *European symposium on research in computer security*, 2015.

[70] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016.

[71] "OpenMined Community. Pysyft Library.," [Online]. Available: https://github.com/OpenMined/PySyft. [Accessed 09 May 2021].

[72] "OpenMined Community. Pygrid," [Online]. Available: https://github.com/OpenMined/PyGrid. [Accessed 21 June 2021].

[73] "OpenMined Community. Kotlinsyft.," [Online]. Available: https://github.com/OpenMined/KotlinSyft. [Accessed 21 June 2021].

[74] "OpenMined Community. Swiftsyft," [Online]. Available: https://github.com/OpenMined/SwiftSyft. [Accessed 21 June 2021].

[75] "OpenMined Community. syft.js," [Online]. Available: https://github.com/OpenMined/syft.js. [Accessed 21 June 2021].

[76] D. Heimbigner and D. Mcleod, " A federated architecture for information management," in *ACM Transactions on Information Systems (TOIS)*, 1985.

[77] A. MACHANAVAJJHALA, X. HE and M. HAY, " Differential privacy in the wild: A tutorial on current practices & open challenges," in *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017.

[78] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage and F. Beaufays, "Applied Federated Learning: Improving Google Keyboard Query Suggestions," in *arXiv preprint arXiv:1812.02903*, 2018.

[79] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," *arXiv preprint arXiv:1811.04017,* 2018.

[80] C. Dwork and R. Aaron, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science,* vol. 9, pp. 211-407, 2014.

[81] T. K. Landauer, P. W. Foltz and L. Darrell, " An introduction to latent semantic analysis," *Discourse processes,* vol. 25, no. 2-3, pp. 259-284, 1998.

[82] X. Yan, J. Guo, Y. Lan and X. Cheng, "A biterm topic model for short texts," in *Proceedings of the 22nd international conference on World Wide Web*, 2013.

[83] T. MINKA, "Estimating a Dirichlet distribution," 2000.

[84] K. Sharma and T. Suryakanthi, "Smart System: IoT for University," in *International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015.

[85] M. Nati, G. Alexander, A. Hamidreza and H. William, "SmartCampus: A user-centric testbed for Internet of Things experimentation," in *16th International symposium on wireless personal multimedia communications (WPMC)*, 2013.

[86] "EU FP7 ICT WISEBED project," [Online]. Available: http://www.wisebed.eu/. [Accessed 12 February 2017].

[87] [Online]. Available: https://www.microsoft.com/en-us/download/details.aspx?id=52367. [Accessed May 2021].

[88] P. Wightman, W. Coronell, D. Jabba, M. Jimeno and M. Labrador, "Evaluation of Location Obfuscation techniques for privacy in location based information systems," in *IEEE Third Latin-American Conference on Communications*, 2011.

[89] L. Yu, L. Ling and C. Pu, "Dynamic Differential Location Privacy with Personalized Error Bounds," in *NDSS*, 2017.

[90] K. Fawaz and S. K. G., "Location privacy protection for smartphone users," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[91] Y. Lu, "Privacy-preserving Logarithmic-time Search on Encrypted Data," in *NDSS*, 2012.

[92] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Annual international cryptology conference*, Berlin, 2001.

[93] G. Ghinita and R. Rughinis, "An efficient privacy-preserving system for monitoring mobile users: making searchable encryption practical," in *Proceedings of the 4th ACM conference on Data and application security and privacy*, 2014.

[94] "Stanford Crypto," [Online]. Available: https://crypto.stanford.edu/pbc/thesis.html. [Accessed 17 June 2020].

[95] "GMP Libraries for C," [Online]. Available: https://gmplib.org. [Accessed 17 June 2020].

[96] [Online]. Available: https://github.com/MonashCybersecurityLab/HXT. [Accessed 17 June 2020].

[97] [Online]. Available: https://www.cs.cmu.edu/afs/cs/academic/class/15492-f07/www/pthreads.html. [Accessed 20 February 2022].

[98] [Online]. Available: https://www.kaggle.com/uciml/sms-spam-collection-dataset. [Accessed 20 February 2022].

[99] S. Caldas, J. Konečny, H. B. McMahan and T. Ameet, "Expanding the reach of federated learning by reducing client resource requirements," in *arXiv preprint arXiv:1812.07210*, 2018.

[100] S. Vijayarani, M. J. Ilamathi and M. Nithya, "Preprocessing techniques for text mining-an overview," in *International Journal of Computer Science & Communication Networks*, 2015.

[101] A. G. Jivani, "A comparative study of stemming algorithms," in *Int. J. Comp. Tech. Appl*, 2011.

[102] S. Deepika, "Stemming Algorithms, A Comparative Study and their Analysis," in *International Journal of Applied Information Systems (IJAIS) –ISSN*, New York, 2012.

[103] D. Harman, "How effective is suffixing?," *Journal of the American Society for Information Science,* vol. 42, pp. 7-15, 1991.

[104] M. F. Porter, "An algorithm for suffix stripping," *Program,* 1980.

[105] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and D. Jeff, Advances in neural information processing systems, 2013.

[106] K. W. Church, "Word2Vec," *Natural Language Engineering,* vol. 23, no. 1, pp. 155-162, 2016.

[107] "Zenodo," [Online]. Available: https://zenodo.org/. [Accessed May 2020].

[108] "Google Scholar," [Online]. Available: https://scholar.google.com/. [Accessed May 2020].

[109] [Online]. Available: https://radimrehurek.com/gensim/models/word2vec.html. [Accessed May 2020].

[110] T. K. LANDAUER, P. W. FOLTZ and D. LAHAM, "An introduction to latent semantic analysis," *Discourse Processes,* vol. 25, no. 2-3, pp. 259-284, 1998.

[111] [Online]. Available: http://www.bom.gov.au/jsp/ncc/climate_averages/solar-exposure/index.jsp?period=an#maps. [Accessed May 2020].

[112] [Online]. Available: https://www.giz.de/en/worldwide/17995.html. [Accessed February 2022].

[113] H. Baali, H. Djelouat, A. Amira and F. Bensaali, "Empowering Technology Enabled Care Using IoT and Smart Devices: A Review," *IEEE Sensors Journal,* vol. 18, no. 5, pp. 1790-1809, 2018.

[114] A. Haroon, S. Akram, M. A. Shah and A. Wahid, "E-Lithe: A Lightweight Secure DTLS for IoT," in *IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Toronto, 2017.

[115] A. K. Jain and R. C. Dubes., Algorithms for Clustering Data, Upper Saddle River: Prentice-Hall, Inc., 1988.

[116] "Kernel panic! What are Meltdown and Spectre, the bugs affecting nearly every computer and device?," techcrunch.com, 2018. [Online]. Available: https://techcrunch.com/2018/01/03/kernel-panic-what-are-meltdown-and-spectre-the-bugs-affecting-nearly-every-computer-and-device. [Accessed 14 02 2018].

[117] E. Rogers, "Understanding Buck-Boost Power Stages in Switch Mode Power Supplies," Texas Instruments, 2007.

[118] J. Silva-Martinez, "ELEN-325. Introduction to Electronic Circuits: A Design Approach," 2008. [Online]. Available: http://www.ece.tamu.edu/~spalermo/ecen325/Section%20III.pdf.

[119] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi and M. H. Brendan, "Towards federated learning at scale: System design," in *arXiv preprint arXiv:1902.01046*, 2019.