A Secure Data Dissemination Scheme Supporting Conjunctive Keyword Searchable Encryption for Connected Vehicle Networks

Afonso Pimentel
afonso.pimentel@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2022

Abstract

There is a growing need to ensure the security of communications, and the modern solution to this problem is using cryptography. Classic cryptographic schemes obtain this security by encrypting the data, which becomes unusable for all users who do not have the secret key.

The dynamic nature of vehicular networks requires communications to be carried out with speed. The overhead that classical encryption adds can be enough to make the exchange of messages unfeasible. With that in mind, this work focuses on exploring cryptographic schemes that generate searchable encrypted data and leverage that in the development of efficient protocols in vehicular networks. The idea is that by adding more functionality to encrypted data, we can compensate for the overhead in computation.

In this work, we searched the literature on Searchable Encryption and analyzed implementations of some schemes. In this analysis, we realized that disjunctive schemes (queries with OR) based on Linear Secret Sharing were, in practice, equivalent to conjunctive schemes (queries with AND). We then modified a conjunctive scheme to make disjunctive queries and verified an improvement in the performance of the query verification speed.

We created two simple protocols for vehicular networks that used Searchable Encryption and tested them in a network simulator, NS3. In the simulator, we created a realistic scenario that made use of traces with the movement of cars and network traffic that imitates the behavior of a standard network. We concluded that protocols that send encrypted SE indexes are unfeasible and proposed solutions to it.

Keywords: Searchable Encryption, ABE, IoV, NS3, VANET

1. Introduction

Modern cars have an increasing number of interconnected sensors and actuators are able to generate a lot of data that can be leveraged in an incredible set of applications. Examples of this are lane-keeping assistance and automatic parking.

The natural extension of this concept is to allow cars to share data amongst themselves and the internet to expand the possibilities of vehicular applications. The goal is to construct a network connecting all cars, road infrastructure, sensors, and cloud servers, forming an Internet of Vehicles (IoV) [6]. An advancement like this can improve mobility by making it safer, faster, and more enjoyable. However, the conventional internet protocols cannot be used in this scenario given that the nodes are moving at high speed. To address this concern researchers have been studying ways of making this Internet of Vehicles possible.

One of the main concerns is how to make the communications on this network secure against malicious users. If attackers can tamper with the data, the consequences can be more catastrophic than when dealing with normal internet communications. Commonly, a system is secured using cryptographic protocols that rely on symmetric and asymmetric encryption.

Although these systems are very successful, they are also very strict and make encrypted data practically useless. In recent years, new solutions are being developed to make it possible to operate on encrypted data. The most famous one is fully homophobic encryption [8] which allows users to perform some operations on encrypted data, such as multiplications, additions, and logical operations without leaking the plain text.

Homomorphic encryption is still very slow and it is impractical for most applications [15]. For that reason, in this work, we intend to study a different cryptographic primitive called Searchable Encryption (SE). The idea is to reduce the functionality of homomorphic encryption to increase perfor-

mance since, in SE, the only functionality is query-encrypted data.

The goal of this project is to explore the current SE solutions and build vehicular network protocols that leverage SE. We will then test them in a simulated environment using NS3 to evaluate if they could have real-world applications, all code developed can be found here [1].

2. Background

2.1. Vehicular Ad-Hoc Networks (VANETs)

Researchers soon realized that a new set of applications could be created by connecting vehicles together with road infrastructure using wireless technology. This idea gained a lot of strength with the rise of the Internet of Things (IoT) which allow for all kinds of sensors, actuators, and devices (commonly called things) to be connected to the internet. It is estimated that 46 billion things will be connected to the internet by the end of 2021 [3].

In order to enable V2V (vehicle-to-vehicle) and V2I (vehicle-to-infrastructure), communications researchers started working on top of the concept of Mobile Ad-Hoc Network (MANET). MANETs are decentralized wireless networks that do not depend on external network infrastructure. Instead, each node acts as router forwarding messages according to the protocol established. VANETs have an extra complication because the network topology is highly dynamic. The nodes are always entering and leaving the network as they are moving at high speed.

The main networking protocols being used for VANETs are defined in the IEEE 1609 standard, commonly known as WAVE (Wireless Access in Vehicular Environments), shown in [1]. It uses IEEE 802.11p standard for dedicated short-range communication (DSRC) for the MAC and Physical layer. Higher networking layers are implemented by IEEE 1609.3.

Normally there are two types of DSRC devices: the onboard units (OBUs) which are the mobile nodes of the network that represent the vehicles and roadside units (RSUs) which are stationary nodes connected to the backbone of the internet. In this context, V2V is performed by OBUs communicating in an Ad-hoc manner and V2R is performed between OBUs and RSUs.

Researchers have not agreed on what the architecture of this new network yet should be. But the main idea lies in having a layer of data collection that encapsulates the sensors and vehicles, a networking layer that combines WAVE and 5G for V2X and Ethernet for the (RSUs) and a cloud layer that has servers and applications that operate on that data. An example of such architecture can be seen in [14].

[1] https://github.com/ampimentel/ABEandVANET

2.2. Searchable Encryption

Most modern applications store some kind of user's data in cloud servers. Because, normally, it is important to perform operations on that data (like search or delete) the information is usually kept unencrypted in the servers. This makes the servers a very desirable place for attackers and has been one of the reasons that so many data leaks have happened. If data would be kept encrypted, even if an attacker could get access to the files he would still need to decrypt the information, adding a new layer of security.

Searchable Encryption schemes were created to solve the aforementioned problem. First introduced by Song in [19], the proposed scheme allowed for single keyword queries with client and server sharing the same key. Boneh later proposed a scheme in [4] that allowed querying with asymmetric keys. These two types of schemes are called Searchable Symmetric Encryption (SSE) and Searchable Public-key Encryption (SPE).

Zhou in [21] defined multiple paradigms in the way SPE schemes are structured, such as Public Key Infrastructure (PKI), Functional Schemes, and Certificateless Encryption (CLE).

PKI follows the classical public key structure. The senders generate encrypted data using the receiver's public key and upload it to the server. The authenticity of the key is verified with the use of certificates. The receiver generates trapdoors using the secret key and sends them to the server, which has the ability to check each data entry.

Functional Encryption (FE) is a new paradigm in modern cryptography. In these types of schemes, the key holders do not recover the original message of the cyphertext, but a specific function of the plaintext and the secret key. It is clear that we can use this type of scheme to construct searchable encryption, by embedding the query into the key and defining a function that returns true if the cyphertext keywords match the query.

FE was first formally studied by Boneh, Sahai and Waters in [5] and can be divided into three subclasses, in respect of how the function is constructed, they are Identity Base Encryption (IBE), Attribute Base Encryption (ABE), Predicate Base Encryption (PE). In IBE the function takes only one attribute that is public, it was created by Shamir in [18] and first used for SE purposes by Boneh in [4]. In ABE the function can take multiple attributes that are also public, it was introduced by Sahai and Waters in [17] and expanded by Goyal, Pandey, Sahai, and Waters [10] who created the concept of access structures, logical expressions that can combine the attribute values in

a more complex way (using **AND** and **OR** gates) to determine if the decryption can be performed. If the keywords are embedded in the cyphertext and the query in the key we call it KP-ABE (key policy ABE), the opposite is called CP-ABE (cyphertext policy ABE), normally KP-ABE is used for SE. Finally in PE, the function attributes

The first schemes developed such as [19], [4] did not allow for multiple keyword search, however in practical systems one must be able to search for multiple attributes and combine them. New schemes were developed to cope with that, and allow the attributes to be combined in two ways: Conjunctive and Disjunctive.

Conjunctive Search was achieved early on by Park in [16], which allows the user to search data entries that contain a specific combination of attributes. Disjunctive search is a way of searching for data that contains at least one of the attributes searched for. It was introduced by Katz in [12]

2.3. Cryptographic Definitions

2.3.1 Bilinear Maps

We define a group $\mathbb{G}$ whose elements are of the form (x, y) and obey the following equation $y^3 = x^2 + ax + b$, modulus p, where p is a prime number. In this group we can add and subtract two points P and G and we can also multiply a point by an integer n, such that its hard to find out n only knowing nP and P.

In some Ellipict curves we can define a bilinear map defined as a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_\mathbb{T}$ with the following properties:

1. $e(P, Q + R) = e(P, Q) * e(P, R)$
2. $e(P + S, Q) = e(P, Q) * e(S, R)$

2.3.2 LSSS

We define LSSS as in [13]. In a Linear Secret Sharing Scheme, a secret s is shared over a set of n parties P, such that:

1. The shares of each party form a vector in $\mathbb{Z}_p$.
2. Each party has a corresponding vector in the form $\rho = (\rho_1, \ldots \rho_n)$. We can then form a matrix M where the row i is the vector ρ from the ith party. We generate the share by selecting a vector $v = (s, r_1, \ldots, r_n)$ where $r_1, \ldots, r_n$ are random numbers, $(Mv)_i$ is the share of the ith party

A set of parties $P_1, \ldots, P_N$ is part of the authorized set if and only if there exists $w_1, \ldots w_N \in \mathbb{Z}_p$ such that $\sum^N w_i * \rho_i = (1, 0, \ldots, 0)$.

2.3.3 General Algorithms in ABE SE

An ABE SE scheme has to provide an implementation for the following algorithms:

- Key Generation: takes as input the system parameters and outputs the keys (sk, pk) for the client
- Encrypt: takes as input a keyword set $W = w_1, \ldots, w_n$ and pk and outputs K
- GenTrap: takes as input a query $Q = q_1, \ldots, q_m$ and the secret key sK and produces a trapdoor T_Q
- Test: takes as input the keyword for a given data entry K and for T_Q and returns true if a capability matches the keywords of a given document

In this work, we are going to evaluate the schemes by comparing the performance of each algorithm. We focus our efforts on studying the last three because they are the most critical for a normal SE application.

3. Implementation

3.1. Searchable Encryption

In this chapter, we will study the implementations of three Searchable Encryption schemes. Two of them are ABE schemes and were implemented by [20], however some changes needed to be done to test them in a more realistic scenario. The authors of these schemes claim to offer conjunctive and disjunctive searches, but we will prove that in those schemes the disjunctive searches are reduced to multiple conjunctive ones.

Leveraging that knowledge we will implement the conjunctive scheme [9] and modify it to allow disjunctive searches. In the end, we are going to benchmark the implementations and find functions that describe the time and space complexity of each algorithm.

3.1.1 LSSS in ABE encryption

All ABE schemes analyzed use LSSS to map attributes into parties. To generate a trapdoor for a query we build a matrix M that will generate shares that build the authorized set based on the desired attributes.

A boolean formula can be encoded in the matrix M using the following algorithm provided in [13]:

1. Convert the boolean formula into an access tree, that has **AND**s and **OR**s as its nodes and attribute values as its leafs
2. We label the root node of vector $v = 1$ and set the global counter $c = 1$

3. We go through the tree in a Breadth-first faction, exploring all nodes of a given level before moving to the next level. If the node n_i is an OR we label its children with $n_i.v$. If n_i is an AND we pad $n_i.v$ with zeros in the end until it has length c. We label one children with $n_i.v|1$ and the other with $(0,\ldots,0)|-1$ where $(0,\ldots,0)$ has size c, in the end increment c.

4. At last we pad all vectors with zeros in the end until they have length c.

We can see that an **OR** gate adds a new dimension to the solution space, so when adding a disjunction, the equation $Mw = (1, 0, \ldots, 0)$ does not have a single solution anymore. Moreover there is at least one extra vector w for every **OR** gate in the boolean formula. In order to evaluate if a set of parties belongs to the authorized set we need to test all possible w.

Most of the research papers seem to disregard that we cannot evaluate a disjunctive query with a single execution of the Test algorithm. This happens because in a disjunction we will have at least two sets of authorized parties which need to be tested separately.

3.1.2 Anonymity Notion

In ABE encryption the attributes are not hidden from the server. Although this might be reasonable in some applications in a SE scenario, the server should not be able to know the query of the client.

To get around that, what is normally done is to divide the keywords inside the query info: Keyword names and Keyword Values. The information on the keyword names is exposed to the server while the keyword values remain hidden. For example, in queries such as *"Name = Afonso and Age = 23"* the server would only know that a search for Name and Age was made, but Afonso and 23 are kept secret. This is a weaker notion of anonymity, but the sensitive part of the query is still protected and makes the ABE schemes possible.

We also have to leak extra information. Because the server does not know the LSSS shares of each party (that would leak the keyword values of a cyphertext) the client needs to provide the appropriate weights for every keyword. With that information, the server can reconstruct the query operations between keywords.

Combining these two types of leakages, in a query such as *"Name = Afonso and Age = 23 or Job = Unemployed"* the server would know: *"Name = ? and Age = ? or Job = ?"*

3.1.3 Disjunctive vs Conjunctive ABE

A perfect disjunctive ABE scheme would be comprised of the following properties:

1. Query anonymity: The server can not recover the relations between keywords in the query

2. Query Evaluation Performance: The time complexity in evaluating a query does not depend on the number of "ORs"

As we see from sections 3.1.2 and 3.1.1 LSSS realizations of Searchable Encryption do not have those properties. This means that evaluating a disjunctive query in this type of scheme is equivalent to evaluating multiple conjunctive ones, it has the same leakage and the same performance.

We can construct a disjunctive scheme from a purely conjunctive scheme by splitting a disjunctive query into multiple conjunctive. For a given entry we evaluate all conjunctive queries separately if at least one is true, the disjunctive query is true.4

3.1.4 Performance Evaluation of ABE Schemes

When operating on elliptic curves the big bottleneck in time is on the pairing operation. Pairings are calculated using complex algorithms and the schemes that perform better use less this operation.

We will test two natural disjunctive schemes [7], [20] and one conjunctive [16] that performs disjunctive searches using the technique described in section 3.1.3. They were implemented in python using a special framework for developing cryptosystems, the Charm Framework [2]. It allows for fast implementation while maintaining a good balance with performance. It also provides tools for benchmarking that facilitate the measures in the time and space domain.

Schemes [7], [20] were already implemented by the authors of [20], but we modified them to allow all types of queries. The Charm implementation of [16] was done from scratch by us.

The testing was extensive and done in the following way: first, generate a database with random values for a given number of columns and lines, then construct several random queries for that database, run these queries with and without searchable encryption and compare the results (check the correctness of the implementation), benchmark all the algorithms. The random generator of the simulations is seeded by an input parameter and each measure is done three times with different seeds.

After the simulations, we plot the results and estimate their behavior using linear regression. The results for the computational time can be seen in

Figures 1, 2 3 and Table 1. To evaluate the cyphertext expansion we use the same technique, the results are shown in Figures 4, 5 and Table 5.

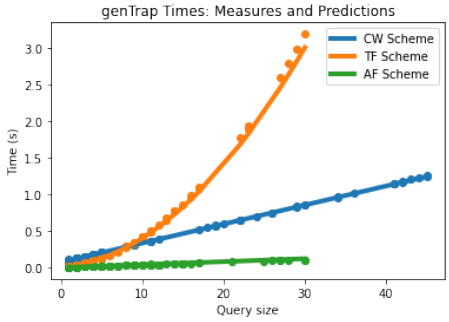

Figure 1: Time measures in seconds for the gentrap algorithm

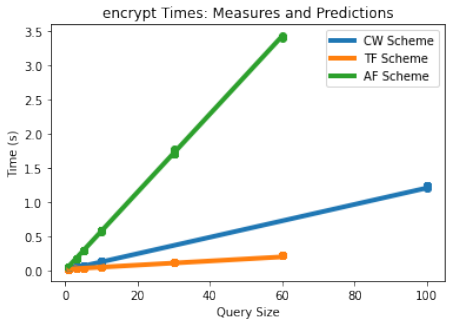

Figure 2: Time measures in seconds for the encrypt algorithm

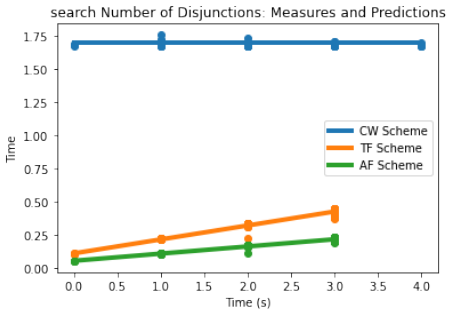

Figure 3: Time measures in seconds for the search algorithm

Sch	GenTrap	Encrypt	Test
CW	$0.026x + 0.07$	$0.012x + 0.01$	$0.326x + 0.07$
TF	$0.003x^2 + 0.01x$	$0.003x + 0.03$	$0.105y + 0.11$
AF	$0.004x$	$0.057x + 0.01$	$0.054y + 0.06$

Table 1: Time in seconds as a function of number of keywords x and number of disjunctions y

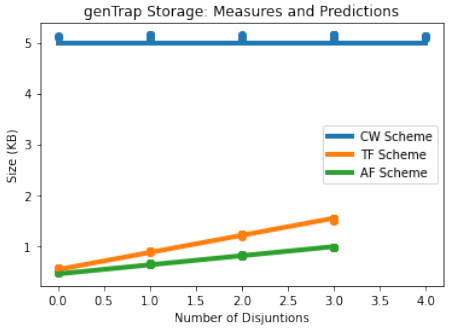

Figure 4: Size of the cyphertext in kilobytes for the gentrap algorithm

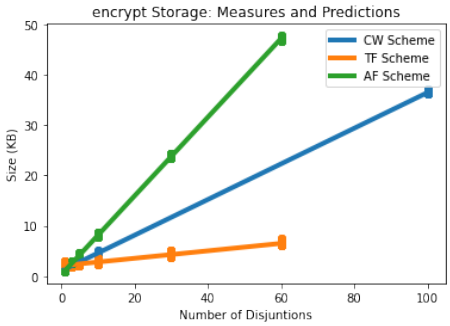

Figure 5: Size of the cyphertext in kilobytes for the encrypt algorithm

Sch	GenTrap	Encrypt
CW	$0.835x + 0.821$	$0.354x + 1.054$
TF	$0.337y + 0.533$	$0.074x + 2.078$
AF	$0.179y + 0.447$	$0.781x + 0.30$

Table 2: Size in kilobytes as a function of the number of keywords x and number of disjunctions y

From the simulations, we conclude that the TF Scheme outperforms the CW in every aspect but the time complexity of the genTrap algorithm. The AF surpasses the TF Scheme in the genTrap and testing phase, but has the worst encryption algorithm amongst all schemes, taking on average 19 times more than the TF algorithm and having a cyphertext 10 times bigger.

Poor performance in the genTrap is usually not a problem because in practical scenarios this algorithm will be executed only once per query. The Test is the most critical algorithm because is going to be run for every line in the database, for every search made. The AF scheme is two times faster when executing a query than the TF scheme. In conclusion, the AF Scheme is better for read-heavy systems, and the TF Scheme is better for write-heavy systems.

3.2. VANET

Now we are going to evaluate, in a vehicular network scenario, the implementations of the schemes [20] and [16] presented in section 3.1 and compare them with their plain text equivalent (no encryption used).

For that we are using the NS3 simulator, we built a scenario that allows for testing application layer protocols in a simple way. This system supports the use of mobility traces to allow the realistic movement of the nodes. After building the scenario, the code collects important metrics and evaluates how the protocols impact the vehicular environment.

In the case of searchable encryption, we are going to build two simple protocols that will allow the evaluation of the most important aspects of SE.

3.2.1 Simulation Environment

What makes a simulation successful is how well it mimics the real world. It is our understanding that two important components need to be addressed in a realistic way, the vehicular physical movement, and the vehicular information movement.

Vehicular physical movement is the way the cars move across the map. A common technique used is making cars move in random ways inside the map. This is not ideal because such movements do not occur in the real world. Better approaches are to use car movement information collected by special sensors or to use traffic simulation tools to simulate real traffic behavior.

Vehicular information movement is about how the data moves in a vehicular network. Most of the simulations done by researchers only have the cars exchange information relevant to the protocol being tested and neglect that in a real network other types of data are being exchanged, impacting the performances of the network and protocol.

For the physical movement, we used SUMO, a traffic simulator tool, to create a double-lane road with cars moving in both directions. We seed this simulation 10 times so we can have different car movements on that road. The scenario is summarized in table 3.

Table 3: Simulation Traces specification

Size (m)	1000×12
Number of cars	58
Duration (s)	150
Maximum speed	$28.0 m/s$
Seeds	$1, 2, 3, 4, 5, 6, 7, 8, 9, 10$

For this project we used the IEEE 802.11p protocol for the inter-vehicle communications, the parameters we used are shown in table 4 and are the same ones described in [11], they are in line with the IEEE 802.11p protocol standard.

Table 4: Wifi 802.11p simulation parameters

Transmitted Power	23 dBm
Transmitted Gain	15 dBi
Received Gain	5 dBi
Frequency	5.9 GHz
Bandwidth	10 MHz

We also install apps in the nodes that simulate the information transmitted in a normal network. First, we install a basic safety message (BSM) app that broadcast a 200 bytes packet with a frequency of 10Hz. This is the elementary application that every vehicular network should have and enables basic safety functionalities to its users.

We simulate extra data exchange in the network by making the nodes transmit UDP packets. To do that we first install an echo UDP server in every node. Then we install a UDP client with a random destination in 10 nodes. Through the simulation, the nodes will transmit UDP packets that will be routed to their destination through the network. Upon receiving a packet, the node will echo the response back to the client. Our goal is to mimic the seemingly chaotic nature of a real network. The parameters of every application are summarized in table 5, the UDP Server does not have a value for the data generation rate because it depends on the number of incoming packets.

	BSM	UDP Server	UDP Client
Apps per Node	0	0	0 or 1
Data Gen Rate (bps)	16000	×	170
Type	Broadcast	Unicast	Unicast

Table 5: Size in kilo bytes as a function of the number of keywords x and number of disjunctions y

4. Results

We created two protocols that leverage searchable encryption. The first one can be seen in 6. A base station is going to communicate with the cars using the radio as a medium. It broadcasts a message containing some public key and some information that the vehicles need to share. When a given vehicle receives this broadcast message, it encrypts the information requested with a searchable encryption scheme, using the public key, and sends it back to the base station.

The second protocol is similar to the first one, but after receiving the broadcast message, the vehicles runs the genTrap algorithm to query the base

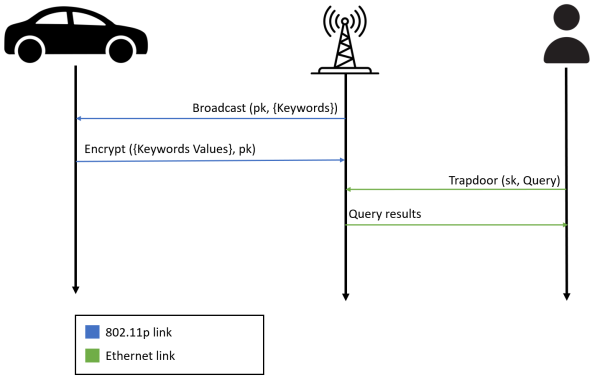

Figure 6: Protocol of the first scenario

station. The BS receives it and does the Test algorithm to find data entries that match the query and send them back to the car. The protocol can be seen in Figure 7.

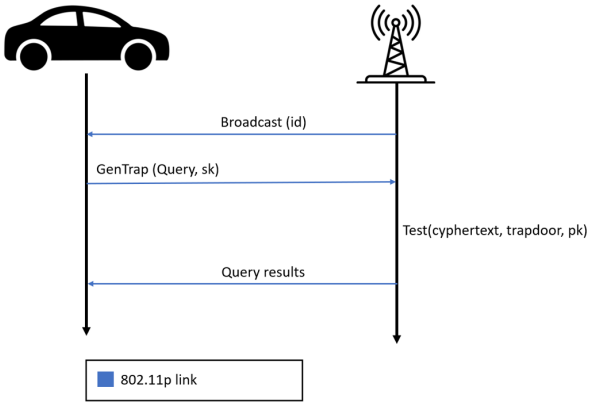

Figure 7: Protocol of the second scenario

Table 6: Simulation Parameters

Simulation Time	100 s
Number of Nodes	58
Base Station Frequency	10 Hz
Vehicle Max Data Gen Rate	32 kbps
Protocol starting time	node id × 0.75
Schemes	Plain Text, [20, 16]
Samples per point	10

After simulating the two protocols, the conclusion we reach is that the schemes perform poorly in the first scenario, to the point that the amount of lost packets probably deems the protocol infeasible in the real world. Protocol 2 yields better results and is achievable if the databases are in the order of 500 entries. Queries should also have limited conjunctions, where 2 might be too much depending on the size of the database.

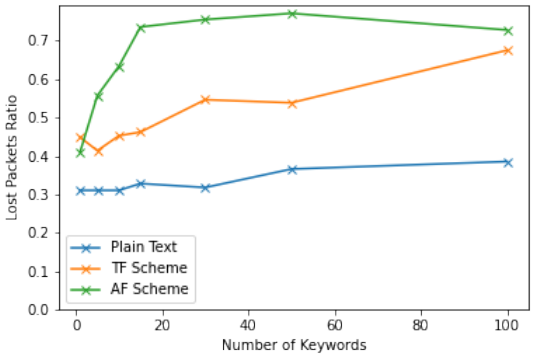

Figure 8: Lost Packets for the first scenario

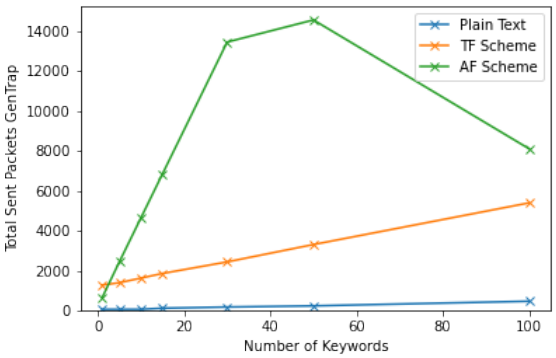

Figure 9: Sent packets for the first scenario

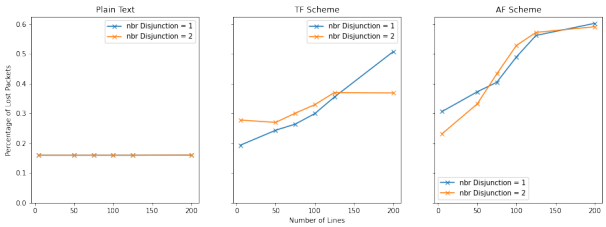

Figure 10: Lost Packets for the second scenario

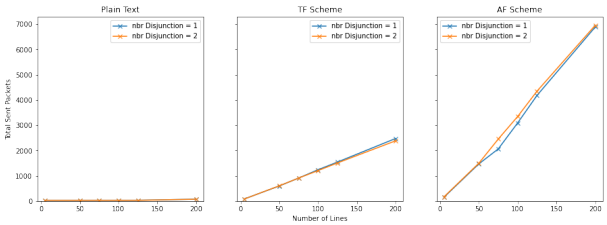

Figure 11: Sent packets for the second scenario

This might seem limiting, but we could create a database that only has certain fields encrypted using SE, we could then create queries that search for secret and nonsecret attributes. We first run the nonsecret query through the database and only

run the secret one with matching elements from the first step. In this way, we could combine the fast querying that modern databases offer with the capabilities of searchable encryption.

The main differences in performance between the 2 scenarios have to do with the number of packets generated by the protocols. Scenario 1 generates much more than scenario 2. We can leverage this knowledge to further improve protocol 2. Since we know that in this scenario the car is the data owner we can keep in the database two encryptions for every entry. One is the keywords generated by the searchable encryption scheme $enc1$, the other can be generated from any other public key cipher available $enc2$. In this way, the base station does the Test algorithm with $enc1$ but returns $enc2$. If the size of $enc2$ is smaller than $enc1$ (probably is) we can achieve much better performance.

5. Conclusions

In this master thesis, we set out to explore searchable encryption and how it could be applied in practical IoV architectures. Currently, security is a concern in vehicular networks and searchable encryption can provide some solutions in that space.

As was discussed, currently SE solutions suffer from being too slow in the Test algorithm and from having a big cyphertext expansion. We verified that the network can tolerate delays in computation, but reacts very badly to the increasingly number of transmitted packets. This means that protocols that rely on transmitting a lot of encrypted keywords via IEEE802.11p are not feasible. Protocols that only need to transmit the results of queries work better because we can combine regular SE schemes with classic public key encryption and transmit only the second one.

5.1. Achievements

In this work, we presented an in-depth practical analysis of the implementations of searchable encryption schemes. Normally SE papers focus more on the theoretical aspects of the schemes and the implementations are only briefly analyzed with superficial tests. Our work tested those implementations in a reliable way by creating meaningful queries with multiple **AND** and **OR**, the queries made on the encrypted database were also done in plain text and the results were compared to test the correctness of the implementations.

We showed that schemes that rely on LSSS to do disjunctive queries were equivalent to a conjunctive scheme doing multiple conjunctive queries. To confirm that, we implemented from scratch a Searchable Encryption Conjunctive scheme [16] and modified it to perform disjunctive queries, these modifications had a Test algorithm twice as fast as the fastest disjunctive scheme we tested.

From the vehicular network standpoint, we built one of the most realistic VANET scenarios using vehicular traces and simulated network traffic. It is also extremely modular and with just a few lines of code, one can simulate a network protocol. With this tool, we tested two different protocols that hinted at what type of situations in searchable encryption could be used best.

5.2. Future Work

For future work, more searchable encryption schemes can be tested, mainly post-quantum ones that rely on a completely different set of assumptions and use different techniques. More scenarios on the IoV architecture can also be explored using different transmission technologies, such as 5G and the complete WAVE stack. New vehicular traces can also be used to have a more complete picture of how the protocols are performing.

Overall, the literature lacks studies that try to estimate the traffic on vehicular networks. Answers to questions such as: "how many applications an average node would use?" and "how much data is exchanged between nodes?" could provide useful insight when building a more realistic network scenario.

References

[1] Ieee guide for wireless access in vehicular environments (wave) architecture. *IEEE Std 1609.0-2019 (Revision of IEEE Std 1609.0-2013)*, pages 1–106, 2019.

[2] J. A. Akinyele, M. D. Green, and A. D. Rubin. Charm: A framework for rapidly prototyping cryptosystems. Cryptology ePrint Archive, Paper 2011/617, 2011.

[3] S. Barker and M. Rothmuller. The internet of things: Consumer, industrial & public services 2020-2024. Technical report, Juniper Research, 2020.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In C. Cachin and J. L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 506–522, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[5] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Y. Ishai, editor, *Theory of Cryptography*, pages 253–273, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[6] J. Contreras-Castillo, S. Zeadally, and J. A. Guerrero-Ibañez. Internet of vehicles: Architecture, protocols, and security. *IEEE Internet of Things Journal*, 5(5):3701–3709, 2018.

[7] H. Cui, Z. Wan, R. H. Deng, G. Wang, and Y. Li. Efficient and expressive keyword search over encrypted data in cloud. *IEEE Transactions on Dependable and Secure Computing*, 15(03):409–422, may 2018.

[8] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 169–178, New York, NY, USA, 2009. Association for Computing Machinery.

[9] P. Golle, J. Staddon, and B. R. Waters. Secure conjunctive keyword search over encrypted data. In *International Conference on Applied Cryptography and Network Security*, pages 31–45, 2004.

[10] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, page 89–98, New York, NY, USA, 2006. Association for Computing Machinery.

[11] R. Halili, M. Weyn, and R. Berkvens. Comparing localization performance of ieee 802.11p and lte-v v2i communications. *Sensors*, 21(6), 2021.

[12] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, pages 146–162, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[13] A. Lewko and B. Waters. Decentralizing attribute-based encryption. In K. G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, pages 568–588, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[14] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark. Connected vehicles: Solutions and challenges. *IEEE Internet of Things Journal*, 1(4):289–299, 2014.

[15] M. Naehrig, K. Lauter, and V. Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop*, CCSW '11, page 113–124, New York, NY, USA, 2011. Association for Computing Machinery.

[16] D. J. Park, K. Kim, and P. J. Lee. Public key encryption with conjunctive field keyword search. In C. H. Lim and M. Yung, editors, *Information Security Applications*, pages 73–86, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[17] A. Sahai and B. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, pages 457–473, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[18] A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology*, pages 47–53, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[19] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S P 2000*, pages 44–55, 2000.

[20] Y.-F. Tseng, C.-I. Fan, and Z.-C. Liu. Fast keyword search over encrypted data with short ciphertext in clouds. Cryptology ePrint Archive, Report 2021/974, 2021.

[21] Y. Zhou, N. Li, Y. Tian, D. An, and L. Wang. Public key encryption with keyword search in cloud: A survey. *Entropy*, 22(4), 2020.