Privacy Preserving Services for Intelligent Transportation Systems with Homomorphic Encryption

Aymen Boudguiga[1], Oana Stan[1], Abdessamad Fazzat[2], Houda Labiod[2] and Pierre-Emmanuel Clet[1]

[1]*Université Paris-Saclay, CEA-List, 91120, Palaiseau, France*

[2]*INFRES, Telecom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France*

Keywords: Privacy, C-ITS, Homomorphic Encryption.

Abstract: With the advent of intelligent transportation systems, vehicles will connect continuously to the Internet via the vehicular core network or the cellular network. Opening vehicles systems to the Internet aims at improving vehicles safety and comfort via the development of remote services for drivers assistance. Such services are for example infotainment applications, software update over the air, remote diagnostics and adaptive insurance. However, some of these services come with an inherent problem of privacy as they require as inputs the private data from the vehicles. In this work, we investigate the use of homomorphic encryption for ensuring the confidentiality of vehicles private data. We study the confidentiality of data, which are treated by external service providers such as cars manufacturers, their stakeholders and insurances. Our protocol ensures, by design, the private treatment of vehicles data thanks to homomorphic encryption properties. We validate our proposal by studying drivers behaviour using a simple neural network that takes as input drivers pictures and tells whether a driver is concentrated or distracted. Indeed, we rely on a 3 layers network for classifying drivers behavior in 10 different classes from normal to dangerous. We use a quadratic activation function for intermediate layers which contain 20 and 10 units, respectively. Meanwhile, we use a sigmoid activation function for the last layer which contains 10 units, one per label. Our classification takes 11 seconds with a classification accuracy of 86% and 25 seconds with a classification accuracy of 92%.

1 INTRODUCTION

Nowadays, transportation systems are evolving towards autonomous driving. They will benefit from wireless car-to-car and car-to-infrastructure connectivities provided by upcoming Cooperative Intelligent Transportation System (C-ITS). The combination of wireless connectivity and automatic driving capabilities is creating new services not only targeting vehicle and driver safety but also providing new infotainment applications. Most of these services rely on collected data about drivers and their vehicles, and raise new challenges for personal privacy.

In this work, we target the privacy of drivers using remote services that collect data from their connected vehicles. The collected data serve to propose a dedicated remote service to the vehicle driver. By data, we refer to vehicle position, acceleration periods, braking frequencies, and in general, all the information that come from the internal network of a vehicle (e.g., electronic controllers, drivers installed applications, vehicle configuration and drivers preferences,...). These data can be exported to external entities such as insurances, Original Equipment Manufacturers (OEM) and their stakeholders or service providers. An OEM can be for example Ford or BMW, a stakeholder entity can be represented by Bosch or Valeo, and a service provider can be Google or Apple as they provide Android Auto and Apple Carplay, respectively.

For example, in the case of a remote service hosted by an insurance company, the collected data from the vehicles serve to adapt insurance fees for a driver if he/she subscribes to a *pay-how-you-drive* service. That is, the insurance will compute a driving profile associated to each driver, using the data collected from his/her vehicle as inputs. Then, the insurance fee is determined by the computed driver profile. As such, a cautious driver will have a reduction of his/her insurance fees, while a dangerous driver will have higher insurance fees.

The collected data from vehicles can also be used by the OEMs and their stakeholders for vehicle remote diagnostics (e-diagnostics) and for proposing

Boudguiga, A., Stan, O., Fazzat, A., Labiod, H. and Clet, P.
Privacy Preserving Services for Intelligent Transportation Systems with Homomorphic Encryption.
DOI: 10.5220/0010349706840693
In *Proceedings of the 7th International Conference on Information Systems Security and Privacy (ICISSP 2021)*, pages 684-693
ISBN: 978-989-758-491-6

software updates over the air. For example, the e-diagnostics service classifies drivers as good or bad with respect to the state of wear of their vehicles. This classification serves to predict possible damages that will happen to vehicles and so, these service providers will adapt accordingly their offer to their clients. In addition, OEMs are interested in collecting data from several vehicles simultaneously to make statistics about breakdowns that touch a common device in their produced vehicles.

The Software update over the air is a critical service as it discloses information about the current version of software installed in vehicles' controllers. This information is too valuable for hackers as it allows them to choose with scrutiny their attacks with respect to the current software vulnerabilities.

Finally, the collected data from vehicles can also be used by vehicular service providers such as Google and Apple to create profiles of drivers. These profiles will then serve to target drivers with personalized applications.

Problem Statement. The major problem of vehicular services is the sensitiveness and the privacy of the collected data from vehicles. For example, this information can be used by a malicious entity to recover drivers home location or their habits of traveling. In addition, current regulation efforts regarding the confidentiality of personal data such as the EU 2016/679 General Data Protection Regulation (GDPR) consider collected data from connected vehicles as private and so their treatment must be confidential and must protect drivers and passengers privacy.

Contribution. In this work, we propose to use homomorphic encryption to ensure the confidentiality of the remote treatment of vehicles private data and so, ensure drivers and passengers privacy. We investigate two deployment options. First, we make each vehicular service provider compute a function over the driver's encrypted inputs, and then return an encrypted result to the concerned driver. Second, we extend the existing connected vehicle architecture with honest-but-curious third parties that will be in charge of homomorphic computation. These third parties will analyse a vehicle private data and return a result to the interested entity (e.g. an OEM or an insurance). That is, the OEMs, the stakeholders or the insurances will never have access to the private data collected from vehicles. Finally, we give some performance indicators about current homomorphic encryption schemes when they are used for drivers data analysis. To do so, we consider as example a remote service for detecting distracted drivers. We rely on an open access dataset with different drivers videos to train a simple neural network for drivers behaviors classification[1]. Once the training is finished, we classify drivers using their encrypted features.

Paper Organization. Section 2.1 reviews the main components of a C-ITS architecture. Section 2.2 defines the basic concepts of homomorphic encryption. Section 2.3 presents the related works on privacy-preserving services for ITS and vehicular networks. Section 3 specifies our protocol for ensuring privacy-preserving vehicular services. Section 4 presents the experiments conducted to validate our proposal and discusses the obtained performance results. Finally, Section 5 concludes the paper with future perspectives and improvements.

2 BACKGROUND

In this section, we first describe the Cooperative Intelligent Transportation System (C-ITS) architecture. Then, we introduce homomorphic encryption. In addition, we review the state of the art on privacy preserving vehicular services. Finally, we present the notations followed in this work.

2.1 ITS Architecture

Figure 1 depicts the components of a Cooperative Intelligent Transportation System (C-ITS). C-ITS relies on two types of access points: Road Side Units (RSUs) installed on roads and On-Board Units (OBUs) embedded in cars. RSUs are gateways to the core network. Meanwhile, OBUs are gateways to vehicles internal networks that connect several Electronic Control Units (ECUs). An OBU serves also as interface to the extra-vehicle network such as 4G/5G, GPS or IEEE802.11p (IEEE standard 802.11p, 2010). The vehicle embedded network is formed by various communication buses such as automotive Ethernet (IEEE Std 100 Base T1, 2018), FlexRay (FlexRay, 2010), MOST (Grzemba, 2011), LIN (ISO-17987-3, 2016) and CAN (Bosch, 1991).

Intelligent vehicles will communicate either with other vehicles (V2V) or with the roadside infrastructure (V2I) using IEEE802.11p or LTE/5G-V2X (IEEE standard 802.11p, 2010; Molina-Masegosa et al., 2020; Sharma et al., 2019). V2X communications provide road safety and improve traffic con-

[1] We use the State Farm dataset provided by Kaggle (https://www.kaggle.com/c/state-farm-distracted-driver-detection). State Farm is an insurance that works on improving alarming to better insure its customers' safety. For example, State Farm studied whether dashboard cameras can help on detecting distracted drivers.

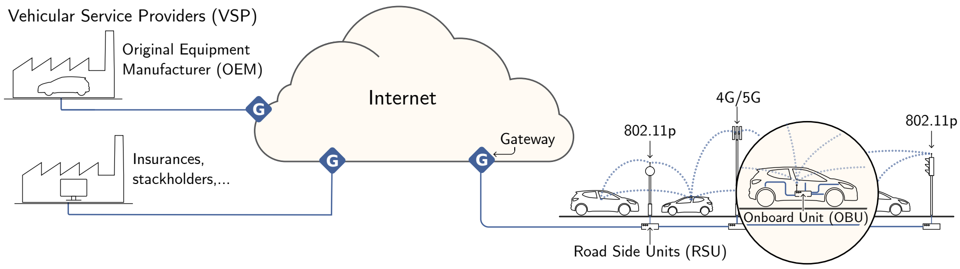

Figure 1: Cooperative Intelligent Transportation System Architecture.

ditions. V2X messages must come from trustworthy parties and only authorized entities can access their contents. Indeed, the standard IEEE1609.2 (IEEE Std 1609.2, 2016) specifies the security requirements for V2X communications. It provides messages integrity and non-repudiation thanks to the use of digital signatures and ensures drivers privacy by using pseudonymous certificates. However, using pseudonymous certificates will not prohibit Vehicular Service Providers (VSP) from accessing drivers personal data at the application level. Indeed, pseudonymous certificates serve mainly to thwart network level attacks such as drivers tracking using V2X messages.

In this work, we demonstrate that homomorphic encryption can be an effective solution to the inherent privacy problem of vehicular services. Indeed, these services are strongly dependant on the driver profile and require access to his/her personal data. Fortunately, homomorphic encryption allows computation over encrypted data and so, can serve to implement privacy-preserving vehicular services. In addition, as vehicular services such as pay-how-you-drive or e-diagnostics do not have hard real-time constraints, the use of homomorphic encryption seems convenient.

Recent works introduced Unmanned Aerial Vehicles (UAV) as part of the C-ITS architecture (Messous et al., 2017; Garg et al., 2018). UAVs provide vision as a service for vehicles. They offer better traffic trajectory and load analysis. In addition, they are supported by the use of Edge servers which extend the C-ITS cloud. Indeed, it is impractical to transmit data from millions of vehicles to data centers in the cloud for processing due to latency and bandwidth constraints. The use of Edge servers ensures faster information analysis, decision making and a quick response to vehicles. That is, instead of carrying their complex computation in a dedicated server on the cloud, intelligent vehicles will offload their computations to the neighboring Edge nodes. The use of Edge servers is advantageous for homomorphic encryption applications, as these nodes can carry out some cumbersome homomorphic operations such as tranciphering, i.e. a cryptographic technique for switching from symmetrically encrypted data to homomorphic data without access to the clear messages (see details below). This allows not only to delegate a part of the computation from the homomorphic computation servers but also to keep a lightweight symmetric encryption on the vehicles side (with usually limited embedded resources) and moreover to save on the network bandwidth.

2.2 Homomorphic Encryption

In 2009, Gentry (Gentry et al., 2009) made a breakthrough in cryptography by proposing the first Fully Homomorphic Encryption (FHE) scheme. That is, Gentry specified a homomorphic encryption scheme E that computes $\mathsf{E}(\mathsf{m}_1 + \mathsf{m}_2)$ and $\mathsf{E}(\mathsf{m}_1 \times \mathsf{m}_2)$ from encrypted messages $\mathsf{E}(\mathsf{m}_1)$ and $\mathsf{E}(\mathsf{m}_2)$. Then, many leveled HE and FHE schemes have been proposed in the literature (Brakerski and Vaikuntanathan, 2011; Brakerski et al., 2012; Fan and Vercauteren, 2012; Van Dijk et al., 2010; López-Alt et al., 2012; Chillotti et al., 2016; Cheon et al., 2016). In practice, a public key homomorphic encryption scheme $\mathsf{HE} = (\mathsf{HE.Keygen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$ is defined by a set of probabilistic polynomial-time algorithms with respect to the security parameter k:

- $(\mathsf{pk}, \mathsf{evk}, \mathsf{sk}) \leftarrow \mathsf{HE.Keygen}(1^{\mathsf{k}})$: outputs an encryption key pk, a public evaluation key evk and a secret decryption key sk. The evaluation key is used during homomorphic operations. This key evk corresponds to the relinearization key in leveled homomorphic schemes such as BFV (Fan and Vercauteren, 2012) or to the bootstrapping key in gate boostrapped schemes such as TFHE (Chillotti et al., 2016).
- $\mathsf{c} \leftarrow \mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{m})$: encrypts a message m into a ciphertext c using the public key pk.
- $\mathsf{m} \leftarrow \mathsf{HE.Dec}_{\mathsf{sk}}(\mathsf{c})$: decrypts a message c into a plaintext m using the secret key sk.
- $\mathsf{c_f} \leftarrow \mathsf{HE.Eval}_{\mathsf{evk}}(\mathsf{f}, \mathsf{c}_1, \ldots, \mathsf{c}_{\mathsf{k}})$: evaluates the function f on the encrypted inputs $\mathsf{c}_1, \ldots, \mathsf{c}_{\mathsf{k}}$ using the evaluation key evk.

Nowadays, we can mix several FHE schemes (e.g. BFV, TFHE, CKKS (Cheon et al., 2016), etc.) using the CHIMERA framework (Boura et al., 2018). As for the overhead induced by the size of the homomorphic ciphertexts during their transmission and storage, we can use transciphering (Canteaut et al., 2015; Albrecht et al., 2016; Hoffmann et al., 2020). This cryptographic technique changes the data encryption algorithm from a classical symmetric encryption to a HE scheme, without decrypting the data. Let m be a plaintext, SYM a symmetric scheme with key k, $\mathsf{SYM.Enc_k(m)}$ the encryption of m with SYM, and HE a homomorphic encryption scheme. With the transciphering, it is enough to run in homomorphic domain the decryption circuit of $\mathsf{SYM.Dec}$ using the homomorphic encryption of the symmetric key $\mathsf{HE.Enc_{pk}(k)}$ to obtain the message encrypted with pk:

$$\mathsf{HE.Eval_{evk}}(\mathsf{SYM.Dec}_{\mathsf{HE.Enc_{pk}(k)}}(\mathsf{SYM.Enc_k(m)})) = \mathsf{HE.Enc_{pk}(m)}$$

Tranciphering is well fitted to the C-ITS architecture as the intelligent vehicles can encrypt their private data using a lightweight symmetric encryption scheme. Then, they outsource the cumbersome homomorphic encryption to a more resource abundant party such as an RSU, an Edge server or a honest-but-curious third party.

As for the application of the homomorphic techniques to the private inference step of neural networks notable works include, in a non-exhaustive way, CryptoNets (Dowlin et al., 2016), DiNN (Bourse et al., 2018), nGraph-HE (Boemer et al., 2018), LOLA (Brutzkus et al., 2019), TAPAS (Sanyal et al., 2018), Faster CryptoNets (Chou et al., 2018). Most of these work are validated over MNIST dataset and none of them addresses our considered use-case.

2.3 C-ITS Privacy State of the Art

The main privacy concern in the ITS context is remaining anonymous and untraceable at the network level (Petit and Kargl, 2018). The standardization solution to this issue is signing V2X messages with short-term pseudonym certificates which are managed by a dedicated Public Key Infrastructure (PKI). Vehicles will change periodically their certificates from a small pool of pseudonyms. A larger pseudonym pool size enhances privacy but weakens efficiency and resistance against Sybil attacks.

Neven et al., (Neven et al., 2017) proposed a generic approach for C-ITS authentication based on privacy-preserving Attribute-Based Credential that generates pseudonyms locally on the vehicle. Their approach enhances the frequency of pseudonym changing while keeping a low exposure against Sybil attacks. Unfortunately, their scheme does not meet the efficiency requirements of real-world C-ITS scenarios such as data latency and bandwidth saturation.

Other solutions have been proposed to ensure location privacy (Kido et al., 2005; Wang et al., 2019; Zhao and Wagner, 2019; Asuquo et al., 2018). Their approaches consist mainly in using anonymization and obfuscation techniques. For example, Ghane et al., (Ghane et al., 2020) addressed the issue of location privacy when the data transportation infrastructure between C-ITS stations is untrusted. They proposed a Differentially Private Data Streaming (DPDS) system. DPDS consists of adding a correlated noise to data before their exchange.

Let us consider now the state of the art regarding the privacy of vehicular services at the application level. In 2011, Troncoso et al., (Troncoso et al., 2011) proposed to install a secure hardware, namely a black box in vehicles to compute the insurance fees locally. The obtained fees are transmitted later to insurances for billing. As such, vehicles private data are kept secret from insurances. The authors also defined a black box auditing mechanism to verify that neither the insurance nor the owner of the vehicle attempted to modify the calculated fees. Indeed, they store the data used to calculate insurance costs on an auxiliary storage. The data are encrypted using a split key between the vehicle owner and the insurance. In case of a dispute, the vehicle owner and the insurance combine their split key to decrypt the auxiliary storage and check how the insurance fee has been computed.

In 2013, Kargl et al., (Kargl et al., 2013) used differential privacy techniques to protect Floating Car Data (FCD). Höfer et al., (Höfer et al., 2013) proposed a privacy preserving charging for electrical vehicles called POPCORN. They relied on group signature and anonymous credentials to enhance the ISO 15118 norm that specifies protocols for smart charging. POPCORN ensure the privacy of vehicles location.

In 2015, Rizzo et al., (Rizzo et al., 2015) proposed a technique to train a decision tree to classify drivers behavior (as aggressive or defensive), while preserving the privacy of collected data and the confidentiality of the decision tree computed by the insurance company. They used a secure version of the ID3 algorithm to build the decision tree by using the homomorphic properties of Paillier's cryptosystem (Paillier, 1999). They also relied on Paillier's cryptosystem homomorphic properties during the classification phase. In 2019, El Ormi et al., (Omri et al., 2019) proposed a privacy preserving *k*-means clustering for driving style recognition. They relied on multiparty computation for computing the distances to clusters centroids. Meanwhile, they used Paillier's cryptosystem during the computation of the new centroids. That is, as for Rizzo et al., they only needed an homomorphic additive scheme to add ciphertexts.

Also an approach using semi-private function evaluation and based on Yao's Garbled circuits is proposed in (Günther et al., 2019) for the calculation of tariff of car insurance companies while hiding the user's data as well as the insurance's private data.

In this work, we benefit from the properties of fully homomorphic encryption to evaluate drivers classification algorithms using a simple neural network with quadratic and sigmoid activation functions.

2.4 Notations

In the following sections, we denote vectors by bold letters, for example $\mathbf{x}$. Each vector $\mathbf{x}$ of n elements can be represented as: $\mathbf{x} = (\mathsf{x}_1, \ldots, \mathsf{x}_\mathsf{n})$. The transpose of a vector $\mathbf{x}$ is denoted $\mathbf{x^t}$. As such the dot product between two vector $\mathbf{x}$ and $\mathbf{y}$ is expressed as: $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x^t} \cdot \mathbf{y}$. We denote by $\mathcal{M}_{m,n}(\mathbb{K})$ the set of matrices with m rows and n columns with entries sampled in $\mathbb{K}$. Matrices are represented by capital letters. $\mathsf{X^t}$ is the transpose of the matrix X.

3 PRIVACY PRESERVING SERVICES FOR C-ITS

In this section, we first present our considered threat model. Then, we specify our privacy preserving protocol for the private treatment of a vehicle's data.

3.1 Threat Model

In this work, we consider a honest-but-curious model (also called the semi-honest model). In this model, many entities $(\mathsf{E}_1, \ldots, \mathsf{E}_\mathsf{n})$, having as secret information $(\mathsf{s}_1, \ldots, \mathsf{s}_\mathsf{n})$, participate to a protocol P to compute a function $\mathsf{F}(\mathsf{s}_1, \ldots, \mathsf{s}_\mathsf{n})$. Each entity $\mathsf{E}_{\mathsf{i}, \mathsf{i} \in [1,\mathsf{n}]}$ is honest and must follow each step of P. However, $\mathsf{E}_{\mathsf{i}, \mathsf{i} \in [1,\mathsf{n}]}$ is curious. That is, $\mathsf{E}_{\mathsf{i}, \mathsf{i} \in [1,\mathsf{n}]}$ will try to find information about other entities secrets $\mathsf{s}_{\mathsf{j}, \mathsf{j} \neq \mathsf{i}}$. P is secure in the honest-but-curious model if each $\mathsf{E}_{\mathsf{i}, \mathsf{i} \in [1,\mathsf{n}]}$ has no other information than $\mathsf{F}(\mathsf{s}_1, \ldots, \mathsf{s}_\mathsf{n})$ at the end of the protocol.

In the honest-but-curious model, the adversary cannot inject modified message as in the Dolev and Yao model (Dolev and Yao, 1981). In addition, using a honest-but-curious adversary avoids message modification attacks against homomorphically encrypted data. Indeed, as homomorphic encryption schemes are malleable by definition, a malicious adversary is able to modify the content of encrypted data.

In this work, vehicle service providers (VSP) will be honest-but-curious. We discuss hereafter two ways for the definition of our privacy preserving service for C-ITS:

1. Each vehicle encrypts its own data using its public key before sending them to the VSP. The VSP will then compute the required output on encrypted data. Finally, the VSP sends the computation result to the vehicle for decryption. It is up to the vehicle to send the decryption result to the VSP.

   One disadvantage of this approach is that each vehicle will have to maintain a pair of public and private keys for homomorphic encryption. That is, we will need to deploy a public key infrastructure dedicated to managing the homomorphic encryption keys needed by millions of vehicles.

2. Each vehicle encrypts its own data using the VSP public key before sending them to a honest-but-curious Homomorphic Computation Server (HCS). The latter is in charge of evaluating the VSP algorithm over the encrypted data. Then, the HCS sends the encrypted output to VSP which recovers the result in clear after decryption. In this case, we assume that the VSP and the HCS do not collude. Indeed, if they collude, they recover the vehicles data. Note that a HCS can perform several homomorphic computations on behalf of different VSP, simultanouesly.

   We prefer this approach because it has the advantage of restricting the deployment of homomorphic encryption keys only to VSPs.

We consider that all vehicles are honest entities. If we consider that vehicles are semi-honest (i.e. honest-but-curious), a vehicle may try a passive attack against the computed function by the HCS. Indeed, a semi-honest vehicle can use the HCS as a computation black box. The semi-honest vehicle provides it with inputs and recovers an associated output. If the computed function is simple, the semi-honest vehicle may attempt to approximate it by computing an interpolation with its stored inputs and their associated outputs.

Finally, we assume that the used homomorphic encryption schemes are well implemented and do not suffer from implementation errors as those pointed by Peng (Peng, 2019). In addition, we focus here on ensuring the confidentiality of data in use and do not consider other security properties such as entities authentication or message integrity validation. We consider that "classical" cryptographic mechanisms are sufficient to provide such properties[2].

3.2 Protocol Description

We specify in this section our protocol for providing a privacy preserving C-ITS service. Our main idea is to add a Homomorphic Computation Server (HCS) to the C-ITS architecture. The HCS does computation on vehicles encrypted data on behalf of different Vehicle Service Providers (VSP).

In Figure 2, we present the four entities participating to our protocol: the vehicle, the Road Side Unit (RSU), the HCS and the VSP.

First, the VSP runs $\mathsf{HE.Keygen}$ and gets pk, evk and sk. Then, it transmits pk and evk to the HCS and the RSU, while the vehicle receives only pk.

The vehicle selects a symmetric key k for a homomorphic-friendly symmetric scheme (Canteaut et al., 2015; Albrecht et al., 2016; Hoffmann et al., 2020). This key is renewed from every protocol run.

In step 2, the vehicle encrypts the symmetric key k using the homomorphic public key $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{k})$. It also encrypts its data $\mathsf{d_v}$ with the symmetric key $\mathsf{SYM.Enc}_{\mathsf{k}}(\mathsf{d_v})$. Then, the vehicle sends $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{k})$ and $\mathsf{SYM.Enc}_{\mathsf{k}}(\mathsf{d_v})$ to the RSU. In step 3, the RSU recrypts $\mathsf{d_v}$ as $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{d_v})$ without access to the vehicle data in clear. Indeed, the RSU applies the transciphering presented in section 2.2 to $\mathsf{SYM.Enc}_{\mathsf{k}}(\mathsf{d_v})$ using $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{k})$. Note that in practice, RSUs are only used for V2X communications. So, it is advantageous to delegate the transciphering to them instead of doing it at the HCS level. As such, we distribute computation over the different players of the protocol. Finally, the RSU sends $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{d_v})$ to the HCS in step 4.

In step 5, the HCS applies the service S to the homomorphically encrypted inputs $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{d_v})$. The obtained result of this analysis $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{r_v})$ is sent to the VSP in step 6. Note that $\mathsf{r_v} = \mathsf{S}(\mathsf{d_v})$. The VSP obtains $\mathsf{r_v}$ after decryption with its own secret key sk (step 7). Based on the value of $\mathsf{r_v}$, the VSP returns an adapted service $\mathsf{s_v}$ to the vehicle (step 8). If the returned $\mathsf{s_v}$ is a confidential information (e.g. an insurance fee), the VSP can encrypt it with the symmetric key k. Of course, in this case, we make the RSU send $\mathsf{HE.Enc}_{\mathsf{pk}}(\mathsf{k})$ to the VSP (red information in Figure 2).

[2]We refer, for example, to message authentication codes to provide message integrity

4 PERFORMANCE

In this section, we present an example of a service for classifying drivers as concentrated or distracted using a dataset of drivers pictures. Such classification can be used during drivers insurance fees computation. We show how to turn it private thanks to the use of homomorphic encryption. The dataset is provided by the State Farm insurance on Kaggle[3]. The dataset specifies 10 different classes of drivers. Concentrated drivers have their 2 hands on the steering wheel and keep their eyes on the road. Meanwhile, distracted drivers are talking on the phone, drinking coffee or doing makeup.

We use the aforementioned dataset to train a 3 layers neural network where the 2 first layers have a quadratic activation function (Figure 3). Meanwhile, the last layer has a sigmoid activation function. The first layer contains 20 units while the second and third layers contain 10 units each. We choose to use the quadratic activation function as it is easily implemented in the homomorphic domain (no need for its approximation by interpolation as for other non-linear activation functions).

4.1 Neural Network Training

We use `Python 3` and the `Tensorflow` framework for the dataset preprocessing and for encoding our neural network (NN) model.

For the training step, we choose 90% of the 22424 labelled records of the dataset randomly. Each entry of this dataset is an RGB picture of the shape $480 \times 640 \times 3$. First, we compress each picture either to $52 \times 52 \times 3$ or to $64 \times 64 \times 3$. Then, we vectorize these 3 dimensional matrices. That is, we represent each picture by a vector of 8112 features or 12288 features, respectively. In addition, we scale all the features by 255. Finally, we train our NN on clear data using a learning rate of 0.0001, minibatches of 32 elements and 100 epochs. We use the sigmoid cross entropy as a cost function.

4.2 Classification of Clear Inputs

Once the training is finished, we run the prediction on the test set of 2243 records (i.e. the remaining 10% of the dataset). The prediction consists in running Algorithm 1 where $\mathsf{W}_{\mathsf{i},\mathsf{i}\in\{1,2,3\}}$ are the matrices of weights and $\mathbf{b}_{\mathbf{i},\mathbf{i}\in\{\mathbf{1},\mathbf{2},\mathbf{3}\}}$ are the biases vectors per layer.

With the NN settings of Section 4.1, we obtain a training accuracy of 88,953% and a classification ac-

[3]https://www.kaggle.com/c/state-farm-distracted-driver-detection

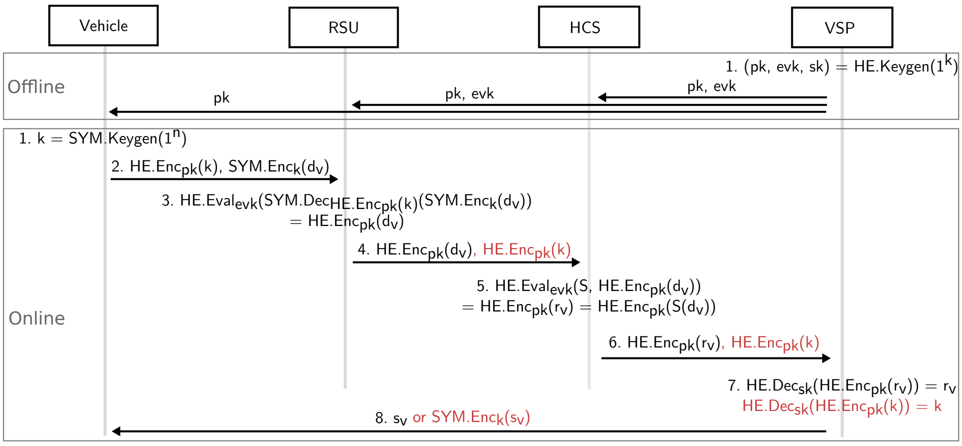

Figure 2: Protocol for providing privacy preserving services for C-ITS.

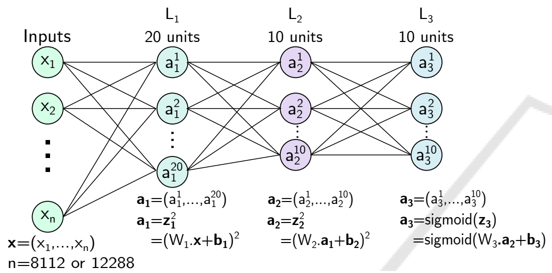

Figure 3: Trained neural network structure.

curacy value of 86,223% for inputs with 8112 features. Accuracy is defined as the ratio between the predicted values and the real ones. In our case, we can improve accuracy by increasing the number of features of the pictures of the dataset. For example, if we compress images to a size of $64 \times 64 \times 3$ (i.e. using 12288 features), we get a training accuracy of 94,247% and a test accuracy of 92,866%.

However, using more features impacts the degree of the cyclotomic polynomial [4] used for homormophic encryption and batching (Smart and Vercauteren, 2011).

4.3 Classification of Encrypted Inputs

In this section, we use the trained NN from section 4.1 to make private classification of input vectors. Indeed, we encrypt the input vectors as they present driver sensitive data. In addition, we encrypt the NN weights and biases as they are private knowledge of the insurance and are shared with the HCS.

[4] We remind that the cyclotomic polynomial $f(x)$ is an irreducible polynomial over $\mathbb{Z}$ which is used to specify the polynomial ring $\mathcal{R} = \mathbb{Z}(x)/(f(x))$. Ciphertexts are polynomials in $\mathcal{R}$.

input : $\mathbf{x}$ a column vector of features, where $\mathbf{x} \in \mathcal{M}_{n,1}(\mathbb{R})$ and $n \in \{8112, 12288\}$
output: $c_{i,i \in [0,9]}$ the class of $\mathbf{x}$

$\mathbf{z_1} = W_1.\mathbf{x} + \mathbf{b_1}$ where $W_1 \in \mathcal{M}_{20,n}(\mathbb{R})$ and $b_1 \in \mathcal{M}_{20,1}(\mathbb{R})$
$\mathbf{a_1} = \mathbf{z_1}^2$
$\mathbf{z_2} = W_2.\mathbf{a_1} + \mathbf{b_2}$ where $W_2 \in \mathcal{M}_{10,20}(\mathbb{R})$ and $b_2 \in \mathcal{M}_{10,1}(\mathbb{R})$
$\mathbf{a_2} = \mathbf{z_2}^2$
$\mathbf{z_3} = W_3.\mathbf{a_2} + \mathbf{b_3}$ where $W_3 \in \mathcal{M}_{10,10}(\mathbb{R})$ and $b_3 \in \mathcal{M}_{10,1}(\mathbb{R})$
$\mathbf{a_3} = \text{sigmoid}(\mathbf{z_3}) = \frac{1}{1+e^{-\mathbf{z_3}}}$
$c_i = \text{argmax}(\mathbf{a_3})$

Algorithm 1: NN prediction algorithm.

To classify a vector $\mathbf{x}$ without revealing any x_i, we first encrypt it as one poylnomial $[x]$ with the API for `CKKS` encryption scheme (Cheon et al., 2016) from Microsoft `SEAL` library. We use batching to accelerate the computation and encode all the x_i in the same polynomial x before its encryption as $[x]$. Batching allows parallel operations in a Simple Instruction Multiple Data (SIMD) fashion (refer to (Smart and Vercauteren, 2011) for details on batching).

Then, we use the updated prediction algorithm 2:

1. In step 1, we encrypt the element of $\mathbf{x}$ in a unique polynomial $[x]$. If we use 8112 features per dataset entry, we specify in `SEAL` a cyclotomic polynomial of degree 16384. As such, we can pack 8192 slots in the same ciphertext polynomial using CKKS. Meanwhile, if we use 12288 features per dataset entry, we specify a cyclotomic polynomial of degree 32768 so that we can pack 16384 slots in the same ciphertext polynomial using CKKS.

   In addition, we encrypt each row $\mathbf{W_1^i}$ of W_1 as a packed polynomial to obtain an encrypted column vector of 20 polynomials $[\mathbf{W_1}] = ([\mathsf{W}_1^1], \ldots, [\mathsf{W}_1^{20}])$. The dot product $[\mathbf{W_1}].[\mathsf{x}]$ results in a column vector of 20 polynomials. The dot product is computed by multiplying $[\mathsf{x}]$ by the polynomial $[\mathsf{W}_1^i], \forall \mathsf{i} \in [1, 20]$ and adding all the slots of the resulting polynomial. Adding all the slots of a batched polynomial requires $\log_2(\mathsf{N}/4)$ rotations and additions, where N is the degree of the chosen cyclotomic polynomial. Finally, we encrypt each element b_1^i of $\mathbf{b_1}$ in a separate packed polynomial $[\mathsf{b}_1^i]$. As such, we obtain the encrypted vector $[\mathbf{b_1}] = ([\mathsf{b}_1^1], \ldots, [\mathsf{b}_1^{20}])$. At the end of step 1, we obtain an encrypted vector of 20 polynomials $[\mathbf{z_1}] = ([\mathsf{z}_1^1], \ldots, [\mathsf{z}_1^{20}])$. Each $[\mathsf{z}_1^i]$ contains in its slots the same encrypted value corresponding to the plaintext $\mathsf{z}_1^i = \mathbf{W_1^i}.\mathbf{x} + \mathsf{b}_1^i$.

2. In step 2, we just square element-wise the encrypted vector of polynomials $[\mathbf{z_1}]$ to get the encrypted vector $[\mathbf{a_1}]$.
3. In step 3, we profit from the format of $[\mathbf{a_1}]$ to compute the dot product $[\mathbf{W_2}].[\mathbf{a_1}]$ in a different way. Indeed, we pack the *column* vectors of W_2 in 20 polynomials. Each column vector has 10 elements and so will use 10 slots per batched polynomial. We multiply the obtained vector $[\mathbf{W_2}]$ element-wise with $[\mathbf{a_1}]$. Then, we add the resulting 20 polynomials together. Finally, we add the obtained result to the batched and encrypted polynomial $[\mathsf{b}_2]$ that corresponds to the vector $\mathbf{b_2}$. We obtain one batched polynomial $[\mathsf{z}_2]$ that encrypts in each of its slots the following plaintext: $\mathsf{z}_2^i = \mathsf{W}_2^i.\mathsf{a}_1 + \mathsf{b}_2^i$ where $\mathsf{i} \in [1, 10]$.
4. In step 4, we just square the encrypted polynomial $[\mathsf{z}_2]$ to obtain the encrypted polynomial $[\mathsf{a}_2]$.
5. In step 5, we use the same method as in step 1 to compute the encrypted vector of polynomials $[\mathbf{z_3}]$. The HCS computes $[\mathbf{z_3}]$ and adds a random noise $\mathbf{e}$ to it in order to avoid leaking information to the VSP when deciphering the NN outputs. The noise vector $\mathbf{e}$ is added in clear. In addition, each of its components must maintain the order of the encrypted values (e.g., $[\mathbf{W_3}].[\mathsf{a}_2] + [\mathbf{b_3}]$). The order is important as the VSP deciphers $[\mathbf{z_3}]$ and gets the class of $[\mathbf{x}]$ by taking the $\mathtt{argmax}(\mathbf{z_3})$. For this work, we took $\mathbf{e}$ as the vector formed by the same random value α repeated 10 times, i.e., $\mathbf{e} = (\mathsf{e}_0 = \alpha, \ldots, \mathsf{e}_9 = \alpha)$. For the futur work, we will investigate other methods for noise setting and leakage avoidance such as interactive encryption as stated in (Boemer et al., 2020).

We implemented Algorithm 2 on an Intel Core i7 and we ran the tests in 1 CPU cadenced at 3,9GHz. The used security parameters respect the default security level provided by `SEAL` which is equal to 128bits. It took 2.368 seconds for data pre-processing and encryption, and 11.664 seconds for the classification of a vector of 8112 features. Meanwhile, it took 4.759 seconds for data pre-processing and 25.667 seconds for the classification of a vector of 12288 features. The difference in classification times depends on the the number of features of the input. Indeed, the number of features dictates the degree of the cyclotomic polynomial to be used for ciphertext representation. In our case, when we used inputs with 8112 features, the degree of the cyclotomic polynomial was 16384. Meanwhile, it was 32768 for inputs with 12288 features. That is, we used bigger ciphertexts to pack inputs with more features, and using bigger ciphertexts results in longer computation times. That is why the classification time was longer for longer input vectors. In addition, we computed the accuracy of the classification with encrypted inputs. We noticed that we got the same classification accuracy as for clear inputs presented in section 4.2.

input : $[\mathbf{x}]$ an encrypted polynomial corresponding to the vector $\mathbf{x} \in \mathcal{M}_{n,1}(\mathbb{R})$
output: $[\mathbf{z_3}]$ an encrypted vector of polynomials corresponding to the logits $[\mathsf{z}_3^i], \mathsf{i} \in [0, 9]$

1 $[\mathbf{z_1}] = [\mathbf{W_1}].[\mathsf{x}] + [\mathbf{b_1}]$
2 $[\mathbf{a_1}] = [\mathbf{z_1}]^2$
3 $[\mathsf{z}_2] = [\mathbf{W_2}].[\mathbf{a_1}] + [\mathsf{b}_2]$
4 $[\mathsf{a}_2] = [\mathsf{z}_2]^2$
5 $[\mathbf{z_3}] = [\mathbf{W_3}].[\mathsf{a}_2] + [\mathbf{b_3}] + \mathbf{e}$

Algorithm 2: NN prediction algorithm with encrypted inputs, weights and biases.

5 CONCLUSION

In this work, we specified a first version of a privacy-preserving protocol for C-ITS services. We made vehicle service providers delegate their computation on vehicle private data to a semi-honest homomorphic computation server. Indeed, the latter is in charge of running services over encrypted data which ensures data confidentiality in use. In addition, we showed that a simple neural network classification of driver behavior using encrypted features and parameters can be effective and may run in reasonable times when we do not have hard real-time constraints. In the future, we plan to:

- study in depth state of the art solutions for applying homomorphic encryption to non-linear activation function such as ReLu. Of course, being able to use other non-linear activation function is important as it can improve the NN accuracy. An interesting solution seems to combine homomorphic encryption with MPC as presented recently in the MP2ML framework (Boemer et al., 2020; Juvekar et al., 2018).

- investigate output layer data randomization to avoid data leakage to the vehicle service provider as discussed by (Boemer et al., 2020; Juvekar et al., 2018).
- improve execution times by investigating hardware algorithmic acceleration for homomorphic schemes such as using GPU for polynomial multiplication acceleration with FFT as proposed by nuFHE[5].

REFERENCES

Albrecht, M., Rechberger, C., Schneider, T., Tiessen, T., and Zohner, M. (2016). Ciphers for mpc and fhe. Cryptology ePrint Archive, Report 2016/687. https://eprint.iacr.org/2016/687.

Asuquo, P., Cruickshank, H., Morley, J., Ogah, C. P. A., Lei, A., Hathal, W., Bao, S., and Sun, Z. (2018). Security and privacy in location-based services for vehicular and mobile communications: An overview, challenges, and countermeasures. *IEEE Internet of Things Journal*, 5(6):4778–4802.

Boemer, F., Cammarota, R., Demmler, D., Schneider, T., and Yalame, H. (2020). Mp2ml: A mixed-protocol machine learning framework for private inference. Cryptology ePrint Archive, Report 2020/721. https://eprint.iacr.org/2020/721.

Boemer, F., Lao, Y., and Wierzynski, C. (2018). ngraph-he: A graph compiler for deep learning on homomorphically encrypted data. *CoRR*.

Bosch (1991). CAN Specification Version 2.0.

Boura, C., Gama, N., Georgieva, M., and Jetchev, D. (2018). Chimera: Combining ring-lwe-based fully homomorphic encryption schemes. Cryptology ePrint Archive, Report 2018/758. https://eprint.iacr.org/2018/758.

Bourse, F., Minelli, M., Minihold, M., and Paillier, P. (2018). Fast homomorphic evaluation of deep discretized neural networks. In *Proceedings of CRYPTO 2018*. Springer.

Brakerski, Z., Gentry, C., and Vaikuntanathan, V. (2012). (Leveled) Fully Homomorphic Encryption Without Bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 309–325.

Brakerski, Z. and Vaikuntanathan, V. (2011). Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524. Springer.

[5]https://github.com/nucypher/nufhe

Brutzkus, A., Oren Elisha, O., and Gilad-Bachrach, R. (2019). Low latency privacy preserving inference. In *Proceedings of the 36th International Conference on MachineLearning, Long Beach, California, PMLR 97*.

Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., and Sirdey, R. (2015). Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. Cryptology ePrint Archive, Report 2015/113. https://eprint.iacr.org/2015/113.

Cheon, J. H., Kim, A., Kim, M., and Song, Y. (2016). Homomorphic encryption for arithmetic of approximate numbers. Cryptology ePrint Archive, Report 2016/421. https://eprint.iacr.org/2016/421.

Chillotti, I., Gama, N., Georgieva, M., and Izabachène, M. (2016). Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22*, pages 3–33. Springer.

Chou, E., Beal, J., Levy, D., Yeung, S., Haque, A., and Fei-Fei, L. (2018). Faster cryptonets: Leveraging sparsity for real-world encrypted inference. *CoRR*.

Dolev, D. and Yao, A. C. (1981). On the security of public key protocols. In *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*, SFCS '81, page 350–357, USA. IEEE Computer Society.

Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., and Wernsing, J. (2016). Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. Technical Report MSR-TR-2016-3.

Fan, J. and Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144.

FlexRay (2010). FlexRay Communications System Protocol Specification Version 3.0.1.

Garg, S., Singh, A., Batra, S., Kumar, N., and Yang, L. T. (2018). Uav-empowered edge computing environment for cyber-threat detection in smart vehicles. *IEEE Network*, 32(3):42–51.

Gentry, C. et al. (2009). Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178.

Ghane, S., Jolfaei, A., Kulik, L., Ramamohanarao, K., and Puthal, D. (2020). Preserving privacy in the internet of connected vehicles. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10.

Grzemba, A. (2011). *MOST-The Automotive Multimedia Network*. Electronics Library.

Günther, D., Kiss, A., Scheidel, L., and Schneider, T. (2019). Poster: Framework for semi-private function evaluation with application to secure insurance rate calculation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2541–2543, New York, NY, USA. Association for Computing Machinery.

Höfer, C., Petit, J., Schmidt, R., and Kargl, F. (2013). Popcorn: Privacy-preserving charging for emobility. In *Proceedings of the 2013 ACM Workshop on Security, Privacy & Dependability for Cyber Vehicles*, CyCAR '13, page 37–48, New York, NY, USA. Association for Computing Machinery.

Hoffmann, C., Méaux, P., and Ricosset, T. (2020). Transciphering, using filip and tfhe for an efficient delegation of computation. Cryptology ePrint Archive, Report 2020/1373. https://eprint.iacr.org/2020/1373.

IEEE standard 802.11p (2010). IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007)*, pages 1–51.

IEEE Std 100 Base T1 (2018). Iso/iec/ieee international standard - part 3: Standard for ethernet - amendment 1: Physical layer specifications and management parameters for 100 mb/s operation over a single balanced twisted pair cable (100base-t1). *ISO/IEC/IEEE 8802-3:2017/Amd 1:2017(E)*, pages 1–92.

IEEE Std 1609.2 (2016). Ieee standard for wireless access in vehicular environments–security services for applications and management messages. *IEEE Std 1609.2-2016 (Revision of IEEE Std 1609.2-2013)*, pages 1–240.

ISO-17987-3 (2016). Road vehicles - Local Interconnect Network (LIN) - Part 3: Protocol specification.

Juvekar, C., Vaikuntanathan, V., and Chandrakasan, A. (2018). Gazelle: A low latency framework for secure neural network inference. *CoRR*, abs/1801.05507.

Kargl, F., Friedman, A., and Boreli, R. (2013). Differential privacy in intelligent transportation systems. In *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*, pages 107–112. ACM.

Kido, H., Yanagisawa, Y., and Satoh, T. (2005). An anonymous communication technique using dummies for location-based services. In *ICPS '05. Proceedings. International Conference on Pervasive Services, 2005.*, pages 88–97.

López-Alt, A., Tromer, E., and Vaikuntanathan, V. (2012). On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1219–1234. ACM.

Messous, M., Arfaoui, A., Alioua, A., and Senouci, S. (2017). A sequential game approach for computation-offloading in an uav network. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–7.

Molina-Masegosa, R., Gozalvez, J., and Sepulcre, M. (2020). Comparison of ieee 802.11p and lte-v2x: An evaluation with periodic and aperiodic messages of constant and variable size. *IEEE Access*, 8:121526–121548.

Neven, G., Baldini, G., Camenisch, J., and Neisse, R. (2017). Privacy-preserving attribute-based credentials in cooperative intelligent transport systems. In *2017 IEEE Vehicular Networking Conference (VNC)*, pages 131–138.

Omri, O. E., Boudguiga, A., Izabachene, M., and Klaudel, W. (2019). Privacy-preserving k-means clustering: an application to driving style recognition. In Liu, Joseph K.and Huang, X., editor, *Network and System Security*, pages 685–696, Cham. Springer International Publishing.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer.

Peng, Z. (2019). Danger of using fully homomorphic encryption: A look at microsoft SEAL. *CoRR*, abs/1906.07127.

Petit, Jonathanand Dietzel, S. and Kargl, F. (2018). *Privacy of Connected Vehicles*, pages 229–251. Springer International Publishing, Cham.

Rizzo, N., Sprissler, E., Hong, Y., and Goel, S. (2015). Privacy preserving driving style recognition. In *Connected Vehicles and Expo (ICCVE), 2015 International Conference on*, pages 232–237. IEEE.

Sanyal, A., Kusner, M., Gascón, A., and Kanade, V. (2018). In *ICML*.

Sharma, V., You, I., and Guizani, N. (2019). Security of 5g-v2x: Technologies, standardization and research directions.

Smart, N. and Vercauteren, F. (2011). Fully homomorphic simd operations. Cryptology ePrint Archive, Report 2011/133. https://eprint.iacr.org/2011/133.

Troncoso, C., Danezis, G., Kosta, E., Balasch, J., and Preneel, B. (2011). Pripayd: Privacy-friendly pay-as-you-drive insurance. *IEEE Transactions on Dependable and Secure Computing*, 8(5):742–755.

Van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. (2010). Fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 24–43. Springer.

Wang, W., Min, M., Xiao, L., Chen, Y., and Dai, H. (2019). Protecting semantic trajectory privacy for vanet with reinforcement learning. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–5.

Zhao, Y. and Wagner, I. (2019). On the strength of privacy metrics for vehicular communication. *IEEE Transactions on Mobile Computing*, 18(2):390–403.