Illuminating the Dark or how to recover what should not be seen

Sergiu Carpov[1], Caroline Fontaine[2], Damien Ligier[1], and Renaud Sirdey[1]

[1] CEA, LIST
Point Courrier 172
91191 Gif-sur-Yvette Cedex
France
[2] LSV, CNRS,
ENS Paris-Saclay,
Université Paris-Saclay,
France

Abstract. Functional encryption (FE) is a cryptographic primitive which allows to partially decrypt ciphertexts, e.g. evaluate a function over encrypted inputs and obtain the output in clear. The downside of employing FE schemes is that some details about input data "leak". We call information leakage of a FE scheme the maximal information one can gain about input data from the clear-text output of FE evaluated function.
FE which are usable in practice support only limited functionalities, in particular linear or quadratic polynomial evaluation. In a first contribution of this work we describe how to combine a quadratic FE scheme with a classification algorithm in order to perform a classification over encrypted data use-case. Compared to direct usage of FE for a linear or a polynomial classifier our method allows to increase classification accuracy and/or decrease the number of used FE secret keys.
In a second contribution we show how to estimate the information leakage of the classification use-case and how to compare it to an ideal information leakage. The ideal information leakage is the minimal information leakage intrinsic to achieve the use-case requirement (e.g. perform a classification task). We introduce a method for estimating the information leakage (real and ideal ones) based on machine learning techniques, in particular on neural networks.
We perform extensive experimentations using MNIST image classification and Census Income datasets. In the case of MNIST, we were able to reconstruct images which are close (in terms of MSE distance and as well as visually) to original images. The knowledge of someones handwriting style facilitate the possibility to impersonate him, to steal his identity, etc. As for the second dataset, we were able to increase the accuracy of predicting input dataset features (e.g. an individual's race) from FE outputs available in clear. Obtained information leakages represent a major security flaw of FE based classifiers because they reveal sensible information about individuals.

Keywords: Functional encryption · Information leakage · Private classification.

Introduction

Functional encryption (FE) is a generalization of traditional public key cryptography. It offers the possibility to partially decrypt ciphertexts with a fine-grained control. More precisely, a FE scheme allows to evaluate a function over encrypted inputs and obtain the output in clear. Cloud computing [6] and verifiable computation [23] are typical fields, among many others, where FE can be used. Functional encryption generalizes attribute-based encryption [27, 19, 26], identity based encryption [5, 24] or predicate encryption [14, 22].

An important application of FE schemes is the classification over encrypted input data. Several works from the literature [21, 25] propose to use simple yet practical FE schemes in order to build classifiers of encrypted images. The inner-product FE (IPFE) was used in [21] and quadratic FE in [25]. In these works, the MNIST dataset of handwritten digits is used to test the realizability and the practical performance of such classifier.

A drawback of FE schemes is that some details about encrypted input data “leak”. We call information leakage of a FE scheme the maximal information one can gain about input data from the clear-text output of the FE evaluated function. Information leakage analysis of classification use-cases based on IPFE was studied in paper [20]. A classifier of MNIST images based on IPFE was used in this work too. The authors used different machine learning algorithms and tools in order to reconstruct input images from the inner-product evaluations given by the FE scheme.

In the the present work we generalize the idea of using a second classifier over linear FE first step from [21] to quadratic FE schemes. We introduce the notion of ideal and real information leakages of FE based use-cases. Roughly speaking, the ideal leakage is the minimal information leakage intrinsic to achieve the use-case requirement (e.g. perform a classification task). Whether the real leakage is the information leakage from an actual implementation of the use-case. The real information leakage is necessarily larger than the ideal one because intermediate use-case data is available in clear form contrary to the ideal one.

A second contribution of this work is the estimation and the comparison of real and ideal information leakages in classification use-cases. Information leakage is estimated using machine learning tools, in particular neural networks. We perform extensive experimental studies on 2 datasets: the MNIST dataset and the Census Income dataset. Compared to the MNIST dataset where an attacker is able to reconstruct (more or less precisely) input digit images, in the case of Census Income dataset an attacker is able to gain more insights on highly-private features of individuals. For example, the attacker will be able to affirm with higher confidence whether an individual is a male or if its race is white, etc.

The MNIST classification based on logistic regression proposed in [25] uses a large number of FE evaluations, at least larger than needed. As we shall see using a second classifier we are able to decrease the number of needed FE evaluations and by consequence to lower the real information leakage.

This paper is organized as follows. We start with a formalization of a generic FE scheme and a description of linear and quadratic schemes. Afterwards in section 2 we describe in details the information leakage in FE based classification use-cases and how to estimate it using machine learning tools. In section 3 we describe the used datasets together with experiments we have performed and we perform an analysis of obtained numerical results about information leakage. Finally, in last section we conclude this work and give some perspectives for future works.

1 Functional encryption

Functional Encryption (FE) is a quite recent generalization of public-key cryptography. This paradigm adds a new party, an *authority*, to the two traditional asymmetric parties. The authority generates a *master secret key* and a *public key*. The master secret key is known only by the authority. This particular key is necessary to derive what is called *secret keys*. The secret keys are associated with functions; for instance, we denote by sk_f the secret key associated with function f. The public key, as expected, is used to encrypt messages. Let ct_x be an encryption of a message x. An user owning sk_f and ct_x can run the evaluation/decryption algorithm and get $f(x)$ as plaintext output. Hence, this is not a traditional way to decrypt, but a kind of evaluation of f over encrypted messages, with an unencrypted result at the end.

Boneh *et al.* give in [6] the following standard definitions for functional encryption using the notion of *functionality*.

Definition 1. *A functionality F defined with (K, X) is a function $F : K \times X \to \Sigma \cup \{\perp\}$. The set K is the key space, the set X is the plaintext space, and the set Σ is the output space and does not contain the special symbol $\perp$.*

Definition 2. *A* functional encryption *scheme for a functionality F is a tuple $\mathcal{FE} = (\texttt{setup}, \texttt{keyGen}, \texttt{encrypt}, \texttt{decrypt})$ of four algorithms with the following properties.*

- *The* `setup` *algorithm takes as input the security parameter 1^λ and outputs a tuple composed of a public key and a master secret key (PUB, MSK).*
- *The* `keyGen` *algorithm takes as inputs the master secret key MSK and $k \in K$ which is a key of the functionality F. It outputs a secret key sk for k.*
- *The* `encrypt` *algorithm takes as inputs the public key PUB and a plaintext $x \in X$. This randomized algorithm outputs a ciphertext c_x for x.*
- *The* `decrypt` *algorithm takes as inputs the public key PUB, a secret key and a ciphertext. It outputs $y \in \Sigma \cup \{\perp\}$.*

It is required that for all $(PUB, MSK) \leftarrow \texttt{setup}(1^\lambda)$, all keys $k \in K$ and all plaintexts $x \in X$, if $sk \leftarrow \texttt{keyGen}(MSK, k)$ and $c \leftarrow \texttt{encrypt}(PUB, x)$ we have $F(K, X) = \texttt{decrypt}(PUB, sk, c)$ with an overwhelming probability.

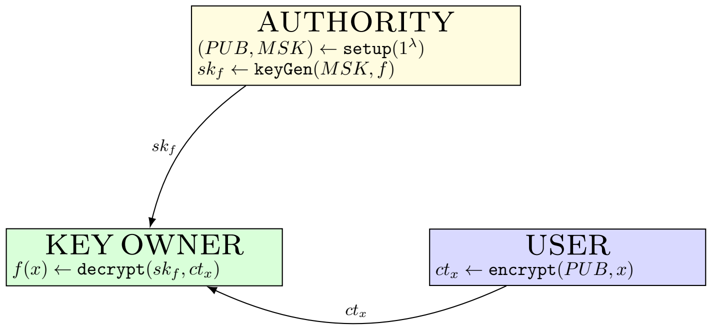

AUTHORITY
$(PUB, MSK) \leftarrow \texttt{setup}(1^\lambda)$
$sk_f \leftarrow \texttt{keyGen}(MSK, f)$
sk_f
KEY OWNER
$f(x) \leftarrow \texttt{decrypt}(sk_f, ct_x)$
USER
$ct_x \leftarrow \texttt{encrypt}(PUB, x)$
ct_x

Fig. 1: The three actors of a functional encryption system, the algorithm they use and their communications. In this figure, the public key, the master secret key and the secret key associated with function f are respectively called PUB, MSK and sk_f.

The cryptographic community is looking for public-key functional encryption schemes enabling to evaluate any polynomial time computable function. Goldwasser *et al.* proposed a construction based on fully homomorphic encryption [11], Garg *et al.* proposed another construction using an indistinguishability obfuscator [10]. At present, however, these constructions remain mostly of theoretical interest. Nevertheless, more recent schemes for simpler functionalities have been proposed, for example FE schemes supporting linear [1, 2] (also called inner-product functional encryption or IPFE) or quadratic [4] polynomial evaluation. The advantage of these schemes is their decent performance and applicability in real applications.

Linear FE or inner-product FE We call linear functional encryption a scheme that enables the evaluation of degree-one polynomials. In the literature these schemes are also called *functional encryption for the inner-product functionality* or *inner-product functional encryption*. Let $\boldsymbol{v}$ be a vector, $ct_{\boldsymbol{v}}$ an encryption of $\boldsymbol{v}$, $\boldsymbol{w}$ be a vector of coefficients, and $sk_{\boldsymbol{w}}$ the secret key associated with $\boldsymbol{w}$. The decryption of ciphertext $ct_{\boldsymbol{v}}$ with secret key $sk_{\boldsymbol{w}}$ returns $\boldsymbol{v}^T \cdot \boldsymbol{w} = \sum_i w_i \cdot v_i$, thus a linear polynomial evaluated at $\boldsymbol{v}$.

Abdalla *et al.* [1] proposed constructions for the inner-product encryption schemes satisfying standard security definitions, under well-understood assumptions like the *Decisional Diffie-Hellman* and *Learning With Errors*. However they only proved their schemes to be secure against *selective adversaries*. Agrawal *et al.* [2] upgraded those schemes to provide them a *full security* (security against *adaptive attacks*).

Quadratic FE We call quadratic functional encryption a functional encryption system that can evaluate degree-two polynomials. A first construction of this type was given in [4]. Let $\boldsymbol{v}$ be a vector, $ct_{\boldsymbol{v}}$ an encryption of $\boldsymbol{v}$, $\boldsymbol{W}$ a square matrix of coefficients and $sk_{\boldsymbol{W}}$ the secret key associated with $\boldsymbol{W}$. The decryption of ciphertext $ct_{\boldsymbol{v}}$ with secret key $sk_{\boldsymbol{W}}$ returns $\boldsymbol{v} \cdot \boldsymbol{W} \cdot \boldsymbol{v}^T = \sum_{i,j} W_{i,j} \cdot v_i \cdot v_j$, thus a quadratic polynomial evaluated at $\boldsymbol{v}$.

2 Information leakage of functional encryption schemes

2.1 Classification use-case

A natural use-case for functional encryption schemes is the classification over encrypted data. Several classification algorithms from the machine learning field use polynomial projection of input data features. As examples we refer to Linear Classification, Gaussian Mixture Model, etc.

Functional encryption schemes found in the literature do not allow to perform arbitrary computations over encrypted data. The schemes which are practical (from a performance point of view) allow to evaluate linear and quadratic polynomials only. Use-cases employing functional encryption need computations which are more complex than these. A solution to bypass this limitation is to adapt the use-case algorithm and decompose it in 2 steps: (i) linear/quadratic polynomial evaluations over encrypted data, (ii) any computation performed on clear data resulting from the first step. Figure 2 illustrates this decomposition. Here, in the first step, k polynomials (given by FE secret keys $sk_{P_1}, \ldots, sk_{P_k}$) are evaluated on encrypted input sample X. Afterwards, as the second step, the relevant additional computations are performed in the clear domain on evaluations $P_1(X), \ldots, P_k(X)$.

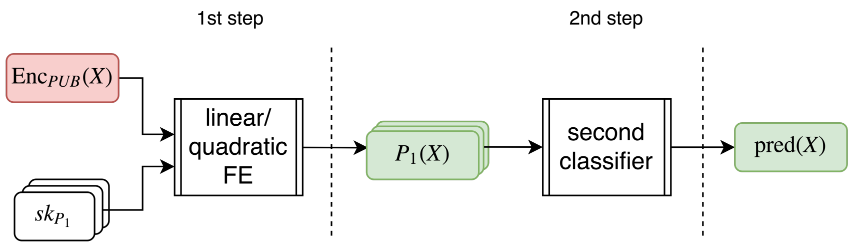


Fig. 2: Algorithm decomposition in a FE use-case. Red and green boxes denote respectively encrypted and clear data.

The classification over encrypted data use-case described above can be built upon this decomposition. The first step of the use-case algorithm is either a linear or a quadratic polynomial evaluation over encrypted data. After this step any computation is possible on the first step output values because they are available in clear form. For the second step we have several possibilities (non-exhaustive list):

– "sign" function for binary classification,

– "arg max" function for one-vs-all multi-class classification,
– a full-fledged classification algorithm.

The last possibility needs some clarification. Any classification algorithm taking as input the polynomial evaluations from the first step can be used as a "final" classification algorithm.

2.2 Information leakage

The output of a FE scheme is clear-text data representing the evaluation of the functionality embedded in the FE secret key. An attacker is able to use this clear text data in order to infer more information about the input data then it is supposed to have. We call *information leakage* this downside of FE schemes. A more formal definition is given below:

Definition 3 (Information leakage). *Let sk_f be a functional encryption secret key embedding functionality f. The* `decrypt` *operation (described in previous section) allows to obtain a clear-text value of $f(x)$ from an encrypted x. The information leakage is the maximal information which can be inferred about x from an access to function's f specification.*

Knowing the specification of f we are able to evaluate f and consequently to obtain clear-text value $f(x)$ for any x. This information leakage definition is straightforwardly generalized to FE scheme instantiations where several secret keys $sk_{f_1}, \ldots, sk_{f_k}$ are available.

In the context of the classification use-case illustrated in Figure 2 we define 2 types of information leakages:

– *ideal* information leakage is the information leakage when only one secret key sk_{pred} is available,
– *real* information leakage is the information leakage when k secret keys $sk_{P_1}, \ldots, sk_{P_k}$ are available.

The ideal leakage corresponds to the case when the entire classification algorithm can be evaluated by the FE scheme. It corresponds to computing both use-case steps without clear-text access to intermediary data. The notion of ideal leakage can be viewed as the minimal information needed to accomplish the use-case requirement, i.e. classifying input samples.

The real information leakage is larger when compared to the ideal one as more clear-text data is available. Measuring the absolute information leakage of a function (for a given input or a given set of inputs) is not easy as it is linked to the Kolmogorov complexity of that input. In essence, the output of a function $f : \{0,1\}^m \longrightarrow \{0,1\}^n$ with $m >> n$ leaks *at most* n bits about the corresponding input (at most because some functions, e.g. a constant function, lead to no leak at all). Intuitively, feeding random data, or equivalently, an input with large Kolmogorov complexity, into any function appears benign because the leak is indeed limited to n bits. Difficulties start to crop up when low Kolmogorov complexity data are fed into a given function because in that case n may not be much smaller than the minimum number of bits required to describe that input. Therefore the induced leak could (in principle, yet intrinsically) allow to retrieve the input in question with enough precision. When considering machine learning algorithms we are typically in a setup where the function purpose in to extract a few highly discriminant bits from highly correlated partially redundant data i.e. data of relatively low Kolmogorov complexity. So we are bending towards the dark side. Yet due to the non-computability of Kolmogorov complexity it is difficult to quantitatively apprehend the above intuitions. Still, the neural network-based leakage estimation technique presented subsequently provides an (efficient) approximation of an oracle able to retrieve low Kolmogorov complexity inputs from the corresponding outputs.

2.3 Leakage estimation

In this section we introduce a method for estimating information leakage of a functional encryption scheme using machine learning tools. We propose to measure the leakage using the information discovery capabilities of a neural network (NN).

Leakage estimation protocol Let f be a functionality encoded in an FE scheme and let X be a representative dataset. By "representative dataset" we understand a dataset which follows the same distribution as a dataset used by the FE scheme. We note that the dataset X is available in clear and that we can straightforwardly obtain $f(x)$ for any $x \in X$. Our goal is to accurately predict/reconstruct dataset samples x from $f(x)$. The accuracy is measured using a suitable metric (more details are given later). To accomplish this goal a neural network is employed. Even though, any machine learning algorithm can be used.

Achieving maximum accuracy is not possible as the problem of obtaining an universal predictor [12, 13, 18] is intractable in the general case. The estimated information leakage (measured by the accuracy metric) will be a lower bound to the real information leakage. Nevertheless, even a such an estimation proves useful for comparing information leakages of different FE instantiations. We make the assumption that the NN model has the same information extraction power independently of FE instantiation it is used upon. This is (supposedly) the case when comparing estimated ideal and real information leakages of classification use-case.

We place ourselves, in the context of the FE-based classification use-case described before. A neural network is used to reconstruct input X from use-case data available in clear. Figure 3 illustrates this methodology. We have 2 possible leakage estimation NNs as a function of use-case clear-text data:

– FE outputs $P_1(X), \ldots, P_k(X)$ (linear/quadratic polynomial evaluations),
– second classifier prediction $\mathrm{pred}(X)$ only.

An estimation of the real information leakage is obtained in the first case and respectively ideal information leakage in the second one. By comparing the prediction accuracies of these 2 NNs we should be able to better understand and to gain insights about classification use-case information leakage.

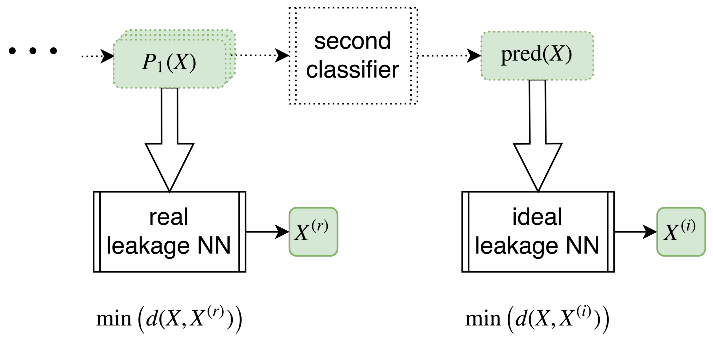


Fig. 3: Real and ideal information leakage estimation using neural network. Top part of this illustration is the FE use-case from figure 2.

3 Experiments

In this section we begin by giving more details about the employed datasets and the neural networks used for classification and for information leakage estimation. Afterwards, we provide an aggregation of the results we obtained. We shall note that no FE schemes evaluation were performed in this work. Typical execution times for linear and quadratic FE schemes decryption (corresponding to polynomial evaluation) is small (few seconds). Please refer to [21, 25] for more details about FE performance.

3.1 Datasets

Two datasets are used in our experimentations. The first one is the well known *MNIST* dataset [17]. The MNIST database is a collection of handwritten digit images. Dataset images have size 28×28 and each pixel has 256 levels of grey. The handwritten digits are normalized in size and are centered. There are 10 output classes in the dataset (digits from 0 to 9). The MNIST dataset has been extensively used by the ML community for classifier validation. For a review of ML methods applied to MNIST, please refer to [7, 16].

The second one is the *Census Income* dataset introduced in [15] and can be found in [3]. This dataset contains highly-sensible personal data (age, education, race, etc.) of approximatively 50 thousands individuals given by 14 features. 6 features are continuous (e.g. age) and other 8 are categorical (e.g. race). Continuous features have been scaled to zero mean and unit variance. Categorical features have been one-hot-encoded, i.e. transformed to binary features denoting whether an individual is in the corresponding category. The prediction task is to determine whether a person earns over 50K$ a year.

These datasets are split into training, validation part and test subsets. The training subset is used for training the prediction classifier and the information leakage estimation neural network. The validation subset is used to choose the final neural network. The test subset is used for asserting the prediction accuracy and respectively the estimated information leakage of the chosen neural network.

3.2 Neural networks structure

We describe in more details the neural networks used for prediction and for information leakage estimation.

Prediction NN The neural network used for prediction has the following structure:

– linear or quadratic first layer,
– one hidden layer (optional),
– one output layer.

First layer The first layer corresponds to the functionality of a linear, respectively quadratic, FE scheme. In this way, the neural network model automatically choses parameters for the secret keys of the FE scheme. This layer has n outputs, i.e. n FE secret keys. In our experiments n belongs to $\{1, \ldots, 10\}$. No activation function is used, so that the hidden and output layers can be directly evaluated from the outputs of a FE scheme.

The linear layer performs simply inner products or equivalently a projection to an n-dimensional space. For the quadratic layer we used the same approximation of quadratic polynomials as in work [25]. In particular, input data is firstly projected to a d-dimensional space for $d \in \{50, 100\}$. Each component of the d-dimensional space is then individually squared. Finally, these components are projected to an n-dimensional space.

Hidden layer The hidden layer is optional. When there is no intermediary layer the neural network corresponds a logistic regression model (or multi-label logistic regression in case of MNIST). In this way we can test simple linear or quadratic prediction models and compare our prediction results with those from [20, 21, 25]. A ReLU (rectified linear unit) activation function is used here. This layer has 256 nodes. Empirical results have shown that there is no need for more than one hidden layer in our experimental setup.

Output layer The output layer has a single node for the Census Income and 10 nodes for the MNIST dataset. The activation functions are sigmoid and respectively softmax. The sign and argmax functions are not included in the neural network to facilitate the training phase.

The validation metric is the accuracy of the obtained neural network classifier.

Information leakage – MNIST The input layer has n nodes which correspond to the FE scheme decryption output. In the case of ideal information leakage estimation the neural network has 10 input nodes (one-hot-encoded digit). Two hidden layers with 256 nodes each are used. The output layer has 784 nodes corresponding to each pixel of the image to reconstruct. The validation metric is the MSE score (mean square error).

Information leakage – Census Income The input layer has n nodes which correspond to the FE scheme decryption outputs. In the case of ideal information leakage estimation the neural network has a single input node. Two hidden layers with 256 and 32 nodes are used. The output layer has a single node.

The information leakage for the Census Income dataset is measured as the capability of the leakage estimation model to make good predictions on a feature from the input dataset. In our experiments we only use binary input dataset features. The input dataset features we use are Sex_Male, Race_White, Race_Black, Race_Asian-Pac-Islander and Race_Other. The validation metric is the ROC AUC score (Area Under the Receiver Operating Characteristic Curve).

To summarize we have 2 datasets, 3 types of FE (one linear and 2 quadratic), 10 possibilities for FE secret keys number and a logistic regression model. Additionally for the Census Income dataset real information leakage estimation is performed on 5 different features. So finally, we have modeled 66 neural networks for the prediction, 198 for real information leakage and 6 for ideal information leakage. Keras [9] framework was used to implement these neural networks. A batched training over 100 epochs is performed. The best network (after each epoch) in terms of optimization metric value on the validation set is chosen as the neural network to keep.

3.3 Results

In this subsection we present the experimental results we have obtained for the prediction and information leakage estimation. In the presented results we use the following notations:

- linear NN model (fe1) – linear FE and a neural network as second classifier,
- linear logistic model (fe1_logit) – linear FE and a logistic regression model as second classifier,
- quadratic NN model (fe2_d) – quadratic FE d-space projection and a neural network as second classifier.
- quadratic logistic model (fe2_d_logit) – quadratic FE with d-space projection and a logistic regression model as second classifier.

MNIST dataset The prediction accuracy (i.e. the ratio between the number of good predictions vs the total number of samples) is given in figure 4. We can observe that there is no significant difference between the projection size of the quadratic models. As expected, the quadratic models always gives better accuracies then the linear ones.

The quadratic logistic model (fe2_d_logit) attains the same accuracy as the quadratic NN model (fe2_d). Although, similar accuracies are obtained by the quadratic NN model starting with 5 FE secret keys. The number of FE decryptions to perform will be smaller in this case.

The linear logistic model accuracy (fe1_logit) corresponds to the accuracy of a linear NN model with $5-6$ FE secret keys.

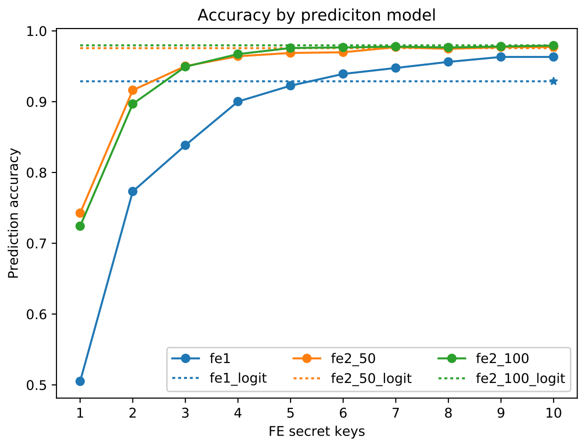


Fig. 4: MNIST dataset – prediction accuracy.

Figure 5 illustrates the information leakage in terms of MSE (lower MSE means more information leaks). The black dotted line is the ideal leakage (Ideal). Observe that the ideal leakage is approximatively equal to the real leakage of linear/quadratic NN models with one FE secret key. We can deduce that one polynomial evaluation reveals the same information about input image as the predicted class of this image.

There is no significant difference between linear/quadratic NN models in terms of information leakage. Although, for bigger number of FE secret keys the linear NN model (f1) leaks a little more information than the quadratic counterpart (fe2_d).

The leakage of logistic models is equivalent with the leakage of linear/quadratic NN models with 10 FE secret keys. This fact was expected as the same number of polynomial evaluations are revealed in both cases.

In order to better see the impact of different number of FE secret keys on the resemblance of reconstructed images with input ones in figure 6 some examples are given. Each image contains 4 lines of digits from 0 to 9 (on rows). First and third lines are input sample images. Second and fourth lines are the reconstructed images with the minimal and respectively maximal MSE score. Thus, second line images are the best reconstructed digits.

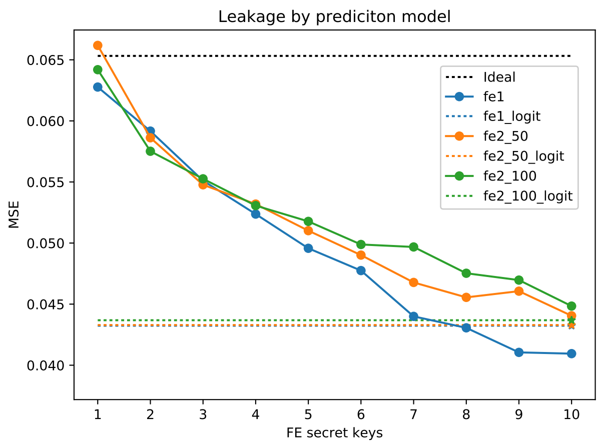


Fig. 5: MNIST – information leakage.

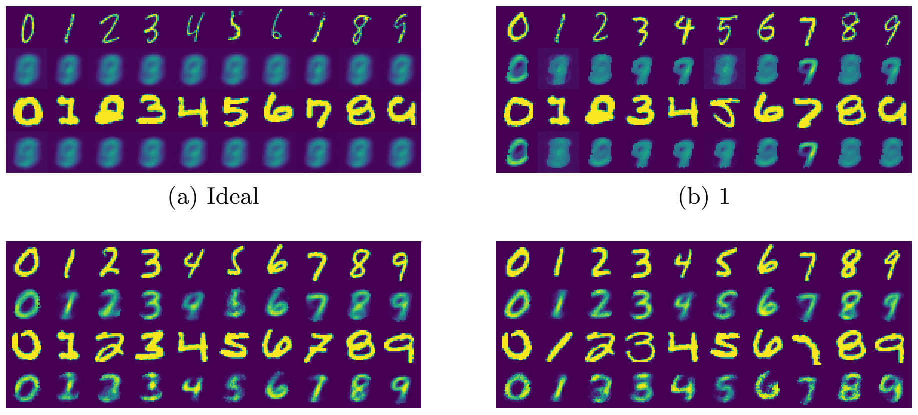
We observe that for the ideal leakage model there is no significant difference between the reconstructed images. On the other side, when 10 FE secret keys are available reconstruction quality is quite good. Reconstructed images for other real information leakage estimation models follow the same resemblance pattern.

Census Income dataset The prediction accuracy obtained for the Census Income dataset is given in figure 7. The logistic models have better performance than the NN ones with the same count (1) of FE secret keys. The NN models become better starting with 2 FE secret keys.

The number of FE secret keys doest not change a lot the classification performances. The fluctuations in the prediction accuracy (after 2 FE secret keys) we observe are due to the fact that the NN learning process is not deterministic. For this particular dataset using a NN model does not significantly change the prediction accuracy of a basic logistic models. Although, using quadratic models with a higher dimensional first projection (100) is better. Sometimes simpler models provide better results.

The bar plot in figure 8 illustrates the ROC AUC score for input dataset feature leakage we study for each prediction model with one FE secret key. We note that a predictor giving an ROC AUC score of 0.5 is equivalent to a random predictor. In our context this means that there is zero information leakage estimated by the NN model. ROC AUC score is never 0.5 because of dataset skewness allowing the NN to perform better than random guessing.

Ideal (blue) and real (red) leakages are plotted on the same bar to ease the comparison. As expected the real information leakage is always larger than the ideal one. It means, for example, that an attacker will be able to increase its confidence in the fact that a given individual is a male.



(a) Ideal (b) 1

(c) 5 (d) 10

Fig. 6: Samples of reconstructed images by the quadratic NN model fe2_100 with different number of FE secret keys (b-d) and the ideal leakage estimation model (a).

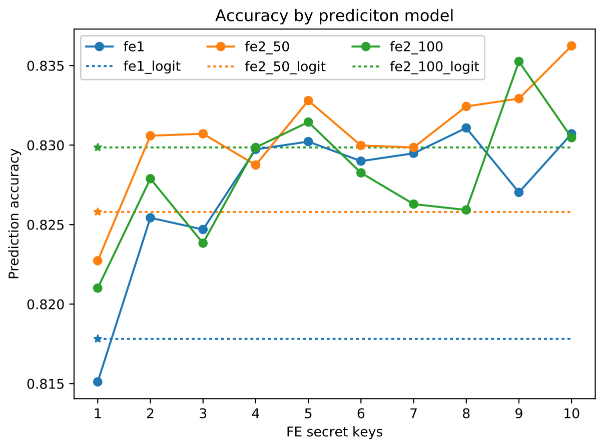


Fig. 7: Census Income – prediction accuracy.

In our experiments we have observed that information leakage estimation depends a lot on the frequency of a studied feature in the input dataset. Information leakages of features which are encountered for a large number of input dataset individuals are better estimated. It is a well know fact that machine learning algorithm learn better on un-skewed datasets.

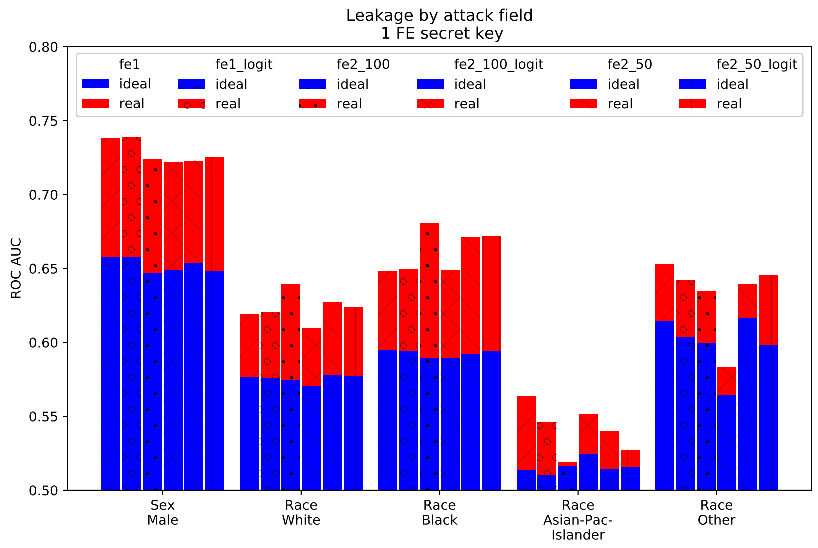


Fig. 8: Census Income – information leakage.

The real information leakage for the MNIST dataset increased with the number of FE secret keys. In order to see if this pattern is verified also for the Census Income dataset we have estimated the leakage of the Sex_Male feature as a function of the number of secret keys. Real and ideal information leakages for this feature are given in figure 9.

As expected the real information leakage increase when the number of FE secret keys increase. The leakage of NN and logistic models is very close for one secret key. When the number of FE secret keys belongs to $9-10$ an attacker will be able to state with very high probability that a given individual is a male. Deploying a prediction model using 9 or 10 FE secret keys will represent a real risk to individual privacy.

Conclusion and future works

In this paper we have studied different FE based classification use-cases and their information leakage. We have combined a basic classifier (based on quadratic FE) with an additional classification algorithm. The use of a second classifier allows

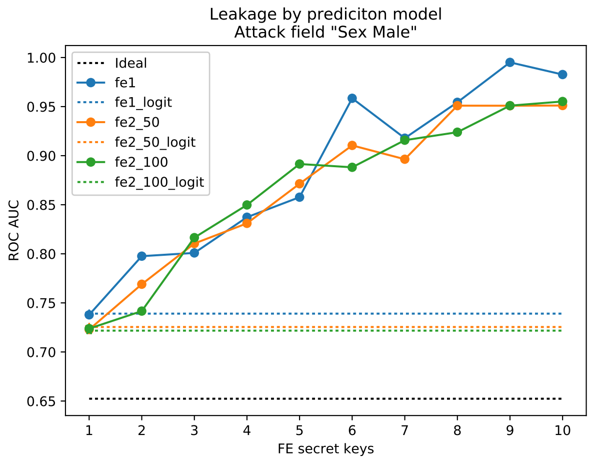


Fig. 9: Census Income – Sex_Male feature information leakage.

either to increase the performance of the overall classification or to decrease the number of FE secret keys. We have proposed a methodology to estimate and compare the information leakage of classification use-cases. Our work is easily generalizable to any FE based use-case.

We have performed extensive experimental studies on classification and information leakage estimation parts. Two well-knows dataset were used, particularly the MNIST digit classification and the Census Income datasets. Experimental result have shown that the information leakage heavily depends on the number of FE secret keys. One way to lower the information leakage is to decrease the number of employed secret keys. Another way is to employ FE schemes with more "complex" functionalities (quadratic FE instead of linear one in our case).

The information leakage was measured by the ability to predict input data set features. We have measured prediction accuracy of input dataset features increase for Census Income. The MSE score between input and reconstructed images was used for MNIST. In future work, we envisage to develop more refined methods to measure the information leakage. For example, in the case of MNIST an image resemblance metric would be more suitable than the MSE.

As a final remark, our work is not limited to linear and quadratic FE schemes only. It can be applied to the more recent FE schemes for polynomial with degree larger than 2 evaluation [8]. Studying the information leakage of use-cases based on these FE schemes will allow to assert more on their operational security in practical applications, and potentially to help finding countermeasures for the kind of information leakage considered in this paper.

References

1. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Simple functional encryption schemes for inner products. In: IACR International Workshop on Public Key Cryptography. pp. 733–751. Springer (2015)
2. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Annual Cryptology Conference. pp. 333–362. Springer (2016)
3. Asuncion, A., Newman, D.: UCI machine learning repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html` (2007)
4. Baltico, C.E.Z., Catalano, D., Fiore, D., Gay, R.: Practical functional encryption for quadratic functions with applications to predicate encryption. In: Annual International Cryptology Conference. pp. 67–98. Springer (2017)
5. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Advances in Cryptology – CRYPTO 2001. pp. 213–229. Springer (2001)
6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Theory of Cryptography Conference. pp. 253–273. Springer (2011)
7. Bottou, L., Cortes, C., Denker, J.S., Drucker, H., Guyon, I., Jackel, L.D., LeCun, Y., Muller, U.A., Sackinger, E., Simard, P., et al.: Comparison of classifier methods: a case study in handwritten digit recognition. In: International conference on pattern recognition. pp. 77–77. IEEE Computer Society Press (1994)
8. Cheon, J.H., Hong, S., Lee, C., Son, Y.: Polynomial functional encryption scheme with linear ciphertext size. Cryptology ePrint Archive, Report 2018/585 (2018), `https://eprint.iacr.org/2018/585`
9. Chollet, F., et al.: Keras. `https://github.com/fchollet/keras` (2015)
10. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on. pp. 40–49. IEEE (2013)
11. Goldwasser, S., Kalai, Y., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Proceedings of the forty-fifth annual ACM symposium on Theory of computing. pp. 555–564. ACM (2013)
12. Hutter, M.: Universal artificial intelligence: Sequential decisions based on algorithmic probability. Springer Science & Business Media (2004)
13. Hutter, M.: On the foundations of universal sequence prediction. In: International Conference on Theory and Applications of Models of Computation. pp. 408–420. Springer (2006)
14. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Advances in Cryptology – EUROCRYPT 2008. pp. 146–162. Springer (2008)
15. Kohavi, R.: Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. pp. 202–207. AAAI Press (1996)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE. vol. 86, pp. 2278–2324. IEEE (1998)
17. LeCun, Y., Cortes, C., Burges, C.J.: The MNIST Database. http://yann.lecun.com/exdb/mnist/
18. Legg, S.: Is there an elegant universal theory of prediction? In: International Conference on Algorithmic Learning Theory. pp. 274–287. Springer (2006)

19. Li, M., Yu, S., Zheng, Y., Ren, K., Lou, W.: Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE transactions on parallel and distributed systems **24**(1), 131–143 (2013)
20. Ligier, D., Carpov, S., Fontaine, C., Sirdey, R.: Information leakage analysis of inner-product functional encryption based data classification. In: PST'17: 15th International Conference on Privacy, Security and Trust. IEEE (2017)
21. Ligier, D., Carpov, S., Fontaine, C., Sirdey, R.: Privacy preserving data classification using inner-product functional encryption. In: ICISSP. pp. 423–430 (2017)
22. Okamoto, T., Takashima, K.: Hierarchical predicate encryption for inner-products. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 214–231. Springer (2009)
23. Parno, B., Raykova, M., Vaikuntanathan, V.: How to delegate and verify in public: Verifiable computation from attribute-based encryption. In: Theory of Cryptography Conference. pp. 422–439. Springer (2012)
24. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 457–473. Springer (2005)
25. Sans, E.D., Gay, R., Pointcheval, D.: Reading in the dark: Classifying encrypted digits with functional encryption. Cryptology ePrint Archive, Report 2018/206 (2018), `https://eprint.iacr.org/2018/206`
26. Waters, B.: Efficient identity-based encryption without random oracles. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 114–127. Springer (2005)
27. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: International Workshop on Public Key Cryptography. pp. 53–70. Springer (2011)