Collusion-resistant Black-box Watermarking in Federated Learning through Weight Relevance Analysis

Elena Rodríguez-Lois, Fernando Pérez-González
Signal Theory and Communications Department
atlanTTic Research Center, University of Vigo, E.E. de Telecomunicación, 36310 Vigo, Spain
erodriguez, fperez@gts.uvigo.es

***Abstract*—Federated Learning (FL) is a promising solution for training machine learning models on data that may contain personal or sensitive information, allowing different data-owners to provide local training updates to a shared model while keeping their data on their own premises. To protect the model from potential misuse or leakage, previous works on FL watermarking have proposed both white-box and black-box schemes, more recently including collusion-resistant traitor tracing capabilities, providing a unique model instance to each data-owner. In case of a leak, the suspected model can be traced back to its origin, even if multiple malicious participants collude. However, there is still a large margin for improving the collusion-resistance capabilities of black-box schemes, where the watermark is embedded into the model's input-output behavior. This work aims at shedding light on this challenging aspect and demonstrates that collusion-resistance can be improved through the identification by the embedder of the relevant weights across different model instances.**

***Index Terms*—DNN watermarking, fingerprinting, federated learning, traitor tracing, Tardos codes, black-box, white-box**

I. Introduction and Previous Works

Federated Learning (FL) has gained popularity in recent years as an alternative to traditional machine learning for applications that train on sensitive or personal data. As these datasets should not be freely disseminated, data-owners collaborate to train a shared model by each performing local training on their own individual datasets. This allows them to share only the updates of this training with a central aggregator, which then integrates them into a shared model which is distributed back to all participants. But even though the training data remains private throughout the process, the shared model itself is still susceptible to espionage, leakage, or unauthorized use from any participant that has access to it.

To deter this kind of malicious behaviour, many prior works have already begun to utilize Deep Neural Network (DNN) watermarking [1] as proof of ownership from the FL system. In many cases, the verification of this watermark requires access to the model weights or hidden activations [2]–[6] (*white-box*), which unfortunately could be impractical in a realistic setting. Because of this, other methods exist that embed the watermark into the input-output behaviour of the model with specific trigger examples, similar to a backdooring attack [7]–[10], making it also accessible through queries to an API (*black-box*), but on the other hand, these usually have a smaller capacity. As both approaches can be complementary, some authors propose to include them together, making for a more robust and reliable accusation [11]–[14].

In some works, individual data-owners watermark the model through their updates [2]–[5], [8], [9], [11], typically to claim their contribution to the shared model.[1] On the other hand, assuming a trusted aggregator, as in this work, they can also embed a watermark, either to protect the shared model as a single object [6], [7], [10], or to identify each participant involved with different watermarks or fingerprints, thus generating slightly different model copies for each data-owner [12]–[14]. The latter is particularly compelling if the watermark also provides traitor tracing capabilities, meaning that the culprits behind a leak could be accurately identified even if different data-owners colluded to create a joint model, as an attack to the uniqueness of their watermarks.

To the best of our knowledge, our previous paper [14] was the first to introduce collusion-resistant black-box fingerprints[2] into the different model copies of the FL framework, identifying each data-owner individually. Results showed that this approach, tested on a simple DNN trained on MNIST [16], could reliably accuse at least one of the guilty participants even after finetuning or pruning for collusions involving up to 6 colluders. While these findings are promising, they also showed that the embedding of black-box triggers was significantly impacted by other data-owners' updates, unlike white-box watermarking. This impact can compromise the traitor tracing capabilities, even when high accuracy is achieved on the trigger set. Unfortunately, this issue is not new to black-box watermarking, and previous works on non-FL models have already discussed that a high trigger accuracy after training does not guarantee any robustness against attacks. However,

Work partially funded by the EU (Next Generation) through the "Plan de Recuperación, Transformación y Resiliencia" under projects FELDSPAR: "Federated Learning with Model Ownership Protection and Privacy Armoring" (Grant MCIN/AEI/10.13039/501100011033) and TRUFFLES: Trusted Framework for Federated Learning Systems (in collaboration with INCIBE), by the Spanish Ministry of Science, Innovation and Universities via a doctoral grant to the first author (FPU22/01929), and by Xunta de Galicia and the European Regional Development Fund, under project ED431C 2021/47.

[1] This could also be used as machine unlearning verification [15].

[2] A white-box client-side collusion-resistant scheme is also proposed in [5].

knowledge of the weights' contribution to the main task can help promote a stronger embedding [17], by avoiding reliance on weak parameters that are more likely to be removed or changed by the attacker. With this in mind, the present work aims to take a step towards developing efficient and reliable schemes for FL black-box traitor tracing. Drawing from our previous work [14] the two main goals are:

- To better understand how the shared updates affect the embedding and its robustness to collusion and how watermarking, in turn, influences these updates.
- To show that even within the FL framework and requirements, we can still pinpoint the relevant weights from the model parameters and use them to strengthen the model's resistance to attacks, including collusion.

The rest of the paper is organized as follows: Section II outlines the proposed black-box approach, Section III describes the technical implementation and discusses the experimental results, and finally, Section IV closes this paper summarizing the main findings.

II. Collusion-Resistant Black-Box Watermarking in Federated Learning

Fingerprinting FL models with unique watermarks for data-owners presents a particular challenge, not considered in the traditional DNN watermarking requirements defined in [1] of *Robustness*, *Security*, *Fidelity*, *Capacity*, *Integrity*, *Generality* and *Efficiency*: As participants have access to not just the final version but many intermediate snapshots of the trained model, copies need to be protected also during training, to prevent a malicious actor from settling for a less than optimal but untraceable model [14]. This new requirement, which could be referred to as *Persistence*, forces both the embedding and the convergence of the main task to stay compatible for many iterations, so solutions that penalize the accuracy of the collusion, such as the recently proposed anti-collusion transformations [18], may not be directly applicable in this case. Instead, our work focuses on embedding collusion-resistant watermarks, able to survive such attacks.

A. Detecting Relevant Weights

A previous work on black-box watermarking [17] found that promoting the embedding into the most relevant weights to the main task significantly improved the robustness of the scheme, as the watermark was memorized through weights that would not change substantially after the attacks. The relevance was determined as the absolute value of the weights after pre-training on the main task. These weights were then exponentially weighted and retrained along with the watermark. In the context of FL, where the aggregator cannot obtain a pre-trained model or a representative dataset for the main task, we propose a similar approach to find the most relevant weights in the intermediate training snapshots, assuming the aggregator has white-box access to the parameters. For any data-owner j and layer l, the kth parameter of the weight vector $\boldsymbol{\theta}^{(l,j)}$ is denoted by $\theta_k^{(l,j)}$ and can be written as $\theta_k^{(l,j)} = \theta_{0k}^l + \delta_k^{(l,j)}$, where θ_{0k}^l is a latent parameter common among all model copies, representing the shared function for the main task, and ensuring gradient compatibility across data-owners. From the point of view of the aggregator, aside from the absolute value $|\theta_k^{(l,j)}|$, one could also hypothesize that the resilience to attacks could be a function of $\delta_k^{(l,j)}$, and weights for which the variance $\sigma^2(\boldsymbol{\theta}_k^l)$ (with σ^2 denoting the variance operator) is small, may be more relevant to the main task.

B. Weight Scaling

Inspired by the work in [17], which enhances the impact of the highest weights by down-scaling the lower ones, this paper explores whether a similar scaling strategy can be used in a FL framework, to promote the use of the most relevant weights in the DNN. As a direct adaptation of [17], considering the absolute value $|\theta_k^{(l,j)}|$, the exponential weighting over the weights of a model copy can be computed as

$$\theta_{\text{EW},k}^{(l,j)} = \frac{\exp\left(|\theta_k^{(l,j)}| \cdot T_{\text{EW}}\right)}{\max_m \exp\left(|\theta_m^{(l,j)}| \cdot T_{\text{EW}}\right)} \theta_k^{(l,j)}, \tag{1}$$

where T_{EW} is a parameter that controls the strength of the scaling. Alternatively, considering the variance across model copies $\sigma^2(\boldsymbol{\theta}_k^l)$, a different scaling approach is

$$\theta_{\text{VS},k}^{l,(j)} = \frac{\exp\left(|\max_m(\sigma^2(\boldsymbol{\theta}_m^l)) - \sigma^2(\theta_k^{l,(j)})| \cdot T_{\text{VS}}\right)}{\max_m \exp\left(\sigma^2(\boldsymbol{\theta}_m^l) \cdot T_{\text{VS}}\right)} \theta_k^{l,(j)}, \tag{2}$$

reducing the impact of the weights with the highest variance, with T_{VS} controlling the strength.

C. Collusion-Resistant Black-Box Scheme in [14]

Although the work in [14] implements both black and white-box fingerprints in an FL framework, building on our earlier paper that introduced these schemes with independent models [19], here we focus exclusively on the black-box approach, as it has proven to be the most challenging to embed and maintain in terms of preserving the traitor tracing capabilities.

To this end, each data-owner j is assigned a vector $\mathbf{x}_j$, that sets a q-ary label x_{ji} for each trigger i in a shared set $\mathcal{T}$ of m triggers, with q being the number of classes in the main task. Vectors $\mathbf{x}_j$ are q-ary Tardos codes [20], sampled from a secret q-component bias vector $\mathbf{p}^{(i)}$ setting the probability of choosing each possible class α, as $P[x_{ji} = \alpha] = p_\alpha^{(i)}$. The bias vector $\mathbf{p}^{(i)}$ is drawn from a symmetric Dirichlet distribution, with $\kappa > 0$ and a cutoff parameter τ, ensuring $p_\alpha \in [\tau, 1 - (q-1)\tau]$ [20].

The accusation is an iterative process, where each example i from the trigger set $\mathcal{T}$ is sequentially queried to the suspected model to obtain the output y_i. At any point in this process, after t triggers have been inputted, an accusation score can be computed for each data-owner j as

$$S_j^t = \sum_{i=1}^{t} S_j^{(i)}, \quad \text{with} \quad S_j^{(i)} = \begin{cases} \sqrt{(1 - p_{y_i}^{(i)})/p_{y_i}^{(i)}} & \text{if } x_{ji} = y_i \\ \sqrt{p_{y_i}^{(i)}/(1 - p_{y_i}^{(i)})} & \text{if } x_{ji} \neq y_i \end{cases}, \tag{3}$$

and an accusation can be made for a participant if any of them surpasses an accusation threshold of

$$Z_t = (\ln \epsilon) \left(-\frac{1}{3 \cdot \sqrt{\tau}} \pm \sqrt{\frac{1}{9 \cdot \tau} - \frac{2t}{\ln \epsilon}} \right), \quad (4)$$

where ϵ is the maximum false positive rate [20]. This means an accusation will not require the full trigger set of m triggers, but rather a subset with cardinality t^*, which ideally will be as small as possible, to keep the process fast, inexpensive and harder to detect. Additionally, as previous queries may be compromised, this approach also saves unused triggers for potential future accusations [19].

For these traitor tracing codes, their traitor tracing capabilities, and thus the value of t^*, will be highly dependent on the so called Marking Assumption (MA). For a collusion of c data-owners, defined by the set of their indexes $\mathcal{C}^c$, the MA holds if the merged model's output y_i to a trigger i is among the labels assigned to the colluders $\mathcal{X}_{\mathcal{C}i}$. Its violation rate (MAV), namely the rate of triggers where $y_i \notin \mathcal{X}_{\mathcal{C}i}$, can also be seen as a measure of the traitor tracing capabilities of the scheme. If the MAV is low, indicating that most triggers provide information about the colluders, the scores in Equation (3) will accumulate sufficient confidence for an accusation, even if some queries do not directly implicate any guilty participants.

III. Experimental Implementation and Results

A. Experimental Setup

Following the implementation in [14], this work considers an image classification task on the MNIST dataset [16], using a shallow network with 3 convolutional layers (with 16, 64 and 128 3×3 kernels) with ReLU activations, followed by a fully connected layer with 10 neurons and a softmax function, where the neuron with the highest value is taken as the predicted class y_i. This simplified model is chosen to facilitate analysis, given the complexity and limited research on black-box traitor tracing in FL. Understanding how these isolated factors affect embedding can guide future development of traitor tracing schemes.

For the main task training, 75% of the MNIST dataset is equally distributed among 100 data-owners, while the rest is reserved for validation and the fine-tuning and pruning attacks. In each round, 10 of them are randomly selected and train locally on a mini-batch of 16 examples with categorical cross-entropy loss. Gradients are averaged at the aggregator, and model copies are updated using SGD with a learning rate of 0.001. The watermarking step is applied before the updated copies are sent back to the data-owners. This process is repeated for 5 epochs (or 1640 iterations).

The configuration of the watermarks was replicated from the work in [14], with a trigger set $\mathcal{T}$ of $m = 1000$ triggers sampled from a uniform distribution in the range [0,1], $\tau = 0.038$, and $\kappa = 100$. The maximum false positive rate for the accusation ϵ is set to 10^{-6}. While this work focuses on black-box fingerprinting, we embed the black-box triggers alongside the white-box watermarks described in [14].

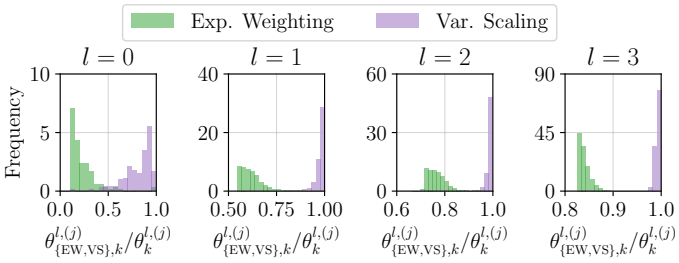


Fig. 1: Scaling factors over trained weights of *Dropout WM*.

The exponential weighting is performed with $T_{EW} = 2$, as done in [17], and the variance scaling uses $T_{VS} = 100$, which makes for a more more subtle transformation. A comparison of how these scaling factors would affect the different layers of the model can be seen in the example presented in Figure 1, calculated over trained weights of the *Dropout WM* strategy.

In these experiments, the collusion attack of $\mathcal{C}^c$ is modeled as an averaging of the model weights across copies from data-owners $j \in \mathcal{C}^c$. Because further attacks can also be performed on the merged model itself, this paper also considers fine-tuning on the validation examples from the MNIST datasets with learning rate of 0.001 and batch size 16 for 5 epochs, and pruning 80% of model weights.

B. Embedding Strategies

At the embedding step, the aggregator is able to perform various strategies to try and improve the traitor tracing capabilities of the scheme. The strategies considered in this work are outlined below.

a) Vanilla WM: The aggregator trains directly on the trigger set $\mathcal{T}$ alongside the white-box regularization in [14], with a mini-batch of 16, learning rate of 0.001, and categorical cross-entropy loss.[3]

b) Dropout WM: The aggregator includes dropout regularization with $p = 0.2$ before every layer of the model.[4]

c) Exp. Weighting DO WM: Adding to *Dropout WM*, exponential weighting is performed before the embedding.

d) Var. Scaling DO WM: Adding to *Dropout WM*, variance scaling is performed before the embedding. Additionally, as comparison, this paper also considers:

e) Independent WM: Independent models trained with batch size 160 (16×10) and embedding after each iteration.[5]

f) No WM: FL models with no embedding at the aggregator, but applying the gradients to different model copies.

C. Experimental Results

The training evolution according to the different embedding strategies is shown in Figure 2. As noted in [14], *Vanilla WM* quickly memorizes the trigger set to achieve perfect accuracy, but loses traitor tracing capabilities as the optimization of both

[3]Our previous work [14] has already shown that this is not enough to ensure traitor tracing capabilities.

[4]Dropout was shown in [14] to promote larger neuron clusters for the trigger set, which appeared to benefit traitor tracing.

[5]Note that this is not an optimal configuration, and it is presented only as a reference for the distribution of model weights without gradient sharing.

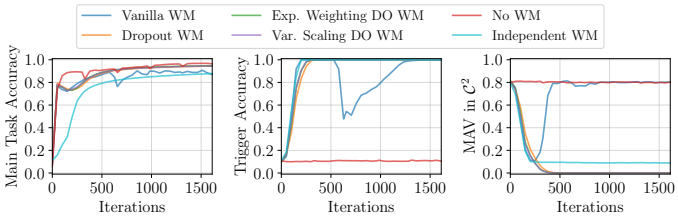
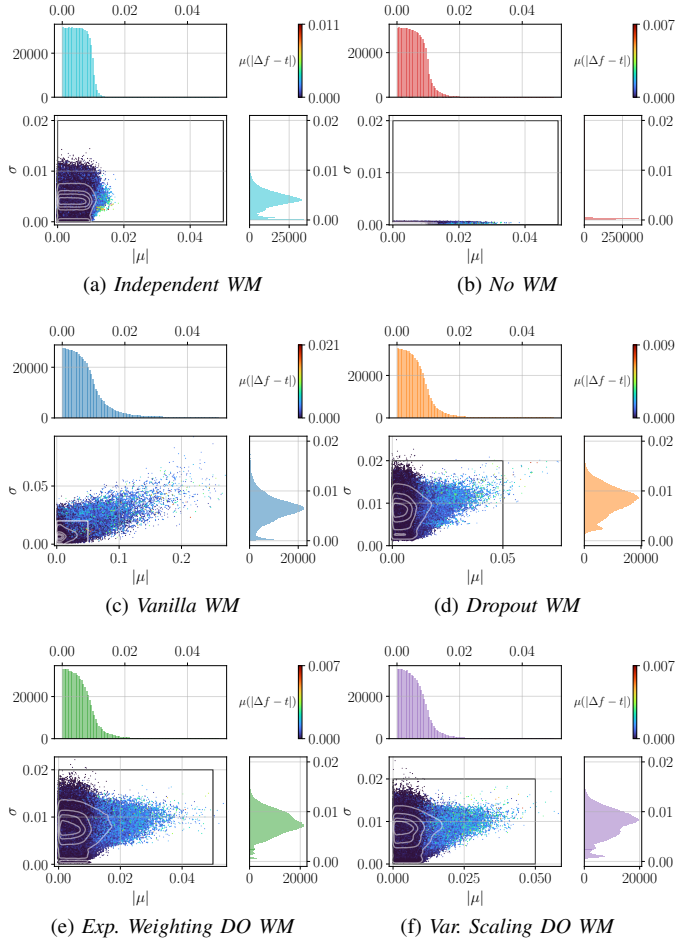


Fig. 2: Average training evolution across 10 random models.



(a) *Independent WM* (b) *No WM*

(c) *Vanilla WM* (d) *Dropout WM*

(e) *Exp. Weighting DO WM* (f) *Var. Scaling DO WM*

Fig. 3: Experimental distribution of weight standard deviation $\sigma(\boldsymbol{\theta}^l)$ and absolute average $|\mu(\boldsymbol{\theta}^l)|$ for the last layer.

tasks confines the watermarking to a small and weak subset of neurons (i.e., no *Persistence*). This also negatively impacts convergence for both tasks, suggesting that the learned $\boldsymbol{\delta}^{(l,j)}$ may lead to slightly incompatible model updates. In contrast, strategies with dropout seem to avoid this issue, achieving accuracies comparable to the *No WM* models, with scaling further improving the evolution of the watermarking task.

Looking into how the different approaches affect the weights $\boldsymbol{\theta}^l$ of the model copies (particularly, for the final layer), Figure 3 compares the distributions in terms of standard deviation $\sigma(\boldsymbol{\theta}^l)$ and absolute value of the average $|\mu(\boldsymbol{\theta}^l)|$, highlighting the average variation after fine-tuning $\mu(|\Delta_{f-t}|)$, as a measure of the distance to a main task local minimum. In

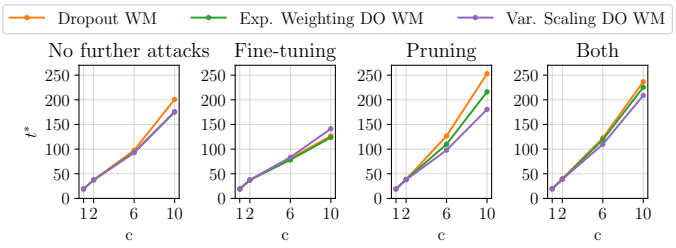


Fig. 4: Experimental t^* according to the embedding strategy.

all cases a rectangle is drawn covering the same area, for which histograms are plotted. As expected, comparing the *Vanilla WM* models (Figure 3c) to the non-watermarked FL models (Figure 3b) reveals a significant difference in weight standard deviation due to the different fingerprints, but the contrast to the distribution of the independently trained watermarked models (Figure 3a) suggests that the increase in both variance and absolute value is likely due to gradients from different model copies leading to non-compatible updates. These outliers are also more affected by the fine-tuning operation, suggesting they may not be well-suited for the main task. Dropout regularization (Figure 3d) and the scaling strategies (Figures 3e and 3f) mitigate this effect, but still there seems to be a tail of outliers susceptible to the fine-tuning.

In terms of the traitor tracing capabilities of the functional FL schemes, Figure 4 presents the average experimental t^* across 500 collusions, according to the embedding strategy, post-collusion attacks and number of colluders $c \in \{1, 2, 6, 10\}$. As noted in Section II-C, an efficient scheme should minimize the number of queries t^*. Aside from the fine-tuning operation, which somewhat mitigates collusion damage, and where the variance scaling does not perform as well, both scaling strategies are improving upon *Dropout WM*, showing that the aggregator can still exploit the weight relevance without explicit knowledge of the main task to achieve a more efficient scheme. Future work will explore a scaling method that optimizes both metrics.

IV. Conclusion

The work presented in this paper is a continuation of our previous work in [14], which showed that traitor tracing watermarking is feasible also for FL models. Though there is still much to understand about the robustness of black-box traitor tracing, this study provides insights into the impact that the shared updates have on the watermarking robustness, and how the different model copies influence the convergence of the main task. Furthermore, the findings suggest that even within the FL constraints and without explicit knowledge of the main task, an aggregator can improve the scheme by appropriately scaling model weights. It is important to note that this work has a limited scope, and the practicality of black-box traitor tracing remains an open question in more challenging scenarios (e.g., non-i.i.d. datasets, a large number of data owners, complex tasks and networks, differential privacy, or new attacks). Future work will address these to allow implementation also in realistic settings.

REFERENCES

[1] Y. Li, H. Wang, and M. Barni, "A survey of deep neural network watermarking techniques," *Neurocomputing*, vol. 461, pp. 171–193, 2021.

[2] J. Liang and R. Wang, "FedCIP: Federated client intellectual property protection with traitor tracking," 2023.

[3] W. Yang, Y. Yin, G. Zhu, H. Gu, L. Fan, X. Cao, and Q. Yang, "FedZKP: Federated model ownership verification with zero-knowledge proof," 2023.

[4] W. Yang, G. Zhu, Y. Yin, H. Gu, L. Fan, Q. Yang, and X. Cao, "FedSOV: Federated model secure ownership verification with unforgeable signature," 2023.

[5] Y. Luo, Y. Li, S. Qin, Q. Fu, and J. Liu, "Copyright protection framework for federated learning models against collusion attacks," *Information Sciences*, vol. 680, no. 121161, 2024.

[6] M. Lansari, R. Bellafqira, K. Kapusta, K. Kallas, V. Thouvenot, O. Bettan, and G. Coatrieux, "FedCrypt: A dynamic white-box watermarking scheme for homomorphic federated learning," 07 2024.

[7] B. G. Atli, Y. Xia, S. Marchal, and N. Asokan, "WAFFLE: watermarking in federated learning," *CoRR*, vol. abs/2008.07298, 2020.

[8] X. Liu, S. Shao, Y. Yang, K. Wu, W. Yang, and H. Fang, "Secure federated learning model verification: A client-side backdoor triggered watermarking scheme," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2414–2419, 2021.

[9] W. Yang, S. Shao, Y. Yang, X. Liu, X. Liu, Z. Xia, G. Schaefer, and H. Fang, "Watermarking in secure federated learning: A verification framework based on client-side backdooring," 2023.

[10] C. Jinyin, M. Li, Y. Cheng, and H. Zheng, "Fedright: An effective model copyright protection for federated learning," *Computers & Security*, vol. 135, p. 103504, 09 2023.

[11] B. Li, L. Fan, H. Gu, J. Li, and Q. Yang, "FedIPR: Ownership verification for federated deep neural network models," 2022.

[12] F.-Q. Li, S.-L. Wang, and A. W.-C. Liew, "Towards practical watermark for deep neural networks in federated learning," 2021.

[13] S. Shao, W. Yang, H. Gu, Z. Qin, L. Fan, and Q. Yang, "Fedtracker: Furnishing ownership verification and traceability for federated learning model," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, pp. 1–18, 01 2024.

[14] E. Rodríguez-Lois and F. Pérez-González, "Exploring federated learning dynamics for black-and-white-box DNN traitor tracing." in *2024 IEEE International Conference on Federated Learning Technologies and Applications (FLTA)*, IEEE, Sept. 2024. (Preprint at https://arxiv.org/abs/2407.02111).

[15] D. M. Sommer, L. Song, S. Wagh, and P. Mittal, "Athena: Probabilistic Verification of Machine Unlearning," 2022.

[16] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[17] R. Namba and J. Sakuma, "Robust watermarking of neural network with exponential weighting," 2019.

[18] H. Cheng, X. Li, H. Wang, X. Zhang, X. Liu, M. Wang, and F. Li, "DeepDIST: A black-box anti-collusion framework for secure distribution of deep models," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 1, pp. 97–109, 2024.

[19] E. Rodríguez-Lois and F. Pérez-González, "Towards traitor tracing in black-and-white-box DNN watermarking with Tardos-based codes," in *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*, IEEE, Dec. 2023.

[20] B. Skoric and J.-J. Oosterwijk, "Binary and q-ary Tardos codes, revisited." Cryptology ePrint Archive, Paper 2012/249, 2012. https://eprint.iacr.org/2012/249.