Re-evaluating the Privacy Benefit of Federated Learning

Mohamed Suliman[1,2], Douglas Leith[1], and Anisa Halimi[2]

[1] Trinity College Dublin, The University of Dublin
doug.leith@tcd.ie
[2] IBM Research Europe - Dublin
{suliman,anisa.halimi}@ibm.com

Abstract. Federated Learning's (FL) main attractive privacy feature of data localisation only holds if FL participants can trust the coordinating server not to carry out data reconstruction attacks, under both honest-but-curious as well as actively malicious threat models. Motivated by our study of the FL system present in Gboard's virtual keyboard, we provide a reassessment of FL's added privacy benefit, and point to three aspects of FL whose affect on privacy requires further research, namely the model architecture, the high levels of trust required to maintain privacy, and vulnerabilities in concrete implementations of the FL protocol.

1 Introduction

Federated Learning (FL) [9] aims to train a machine learning model collaboratively while preserving the privacy of the individual participants. FL provides the privacy benefit of user data not needing to reside on remote servers, while at the same time providing access to pools of otherwise unavailable data, resulting in richer models. This attractive privacy feature of FL has been shown to hold only when the FL participants can *trust* the coordinating server. Several works [5, 6, 11, 14, 17] have demonstrated possible attacks that allow the reconstruction of private user data from observation of individual client updates. Further research has shown [1, 4, 10, 15, 16] that when the server departs from this *honest-but-curious* threat model, and mounts actively malicious attacks against the FL protocol, high fidelity data reconstruction is possible, even under FL hardened with defences such as Secure Aggregation (SA) and Differential Privacy (DP).

The success of these attacks is directly tied to the level of trust required. Specifically, clients need to: (i) trust the server faithfully executes the FL protocol, (ii) trust that the trained models are not designed in a way that allows the reconstruction of user data, (iii) trust the Public Key Infrastructure (PKI) used for secure aggregation, typically managed by the coordinating server, (iv) trust that the server selects a sufficiently large population of users for the current round of FL, without any maliciously added synthetic devices, and (v) trust that the FL software implementation does not enable de-anonymization or bypassing of aggregation.

Taking this inordinate level of trust into account, we ask what is the added privacy benefit of FL as compared to simpler methods of model training? Centralised training with ephemeral data storage (*i.e.* stored only in RAM and deleted immediately after use) appears to be, from a privacy point of view, identical to FL. Both methodologies require the 3rd party orchestrating training to conduct itself in an honest manner. Of course, FL clients can employ Local Differential Privacy (LDP) [12] to add sufficient privacy-preserving noise to their updates before sending them, however, achieving LDP requires a large amount of noise, which degrades the performance of the final learned model [13]. Mitigating this issue requires a large number of clients, a task that once again falls into the hands of the coordinating server. In this paper, we provide a discussion of three aspects of FL that can affect the level of privacy, namely model architecture, the levels of trust involved, and FL system implementations, through the lens of a real world FL system; the Next Word Prediction model present in Google's Gboard virtual keyboard[3]. We argue that these aspects form the foundations of a possible roadmap of future research into FL and privacy.

2 Model Architecture Affects Privacy

Fowl *et al.* [4] introduced the idea of modifying model architecture to improve data reconstruction fidelity. Zhao *et al.* [15, 16] extended this work to develop attacks that can recover up to 80% of the original data with SA of up to 100 clients, and [15] acheive similar levels of privacy leakage for up to 1000 clients. Boenisch *et al.* [1] make use of model parameter modification, an attack much harder to spot than architectural changes, as well as the introduction of synthetic devices to thwart any privacy protection afforded by SA and DP. These synthetic devices can can be made to produce zero gradients, resulting in the recovery of the exact individual gradients of a single, targeted user. Clearly, model architecture affects privacy. With the introduction of Federated Learning as a Service (FLaaS) [7] attacks under this threat model become an increasingly real threat as FLaaS allows for the training of an arbitrary model by an arbitrary 3rd party with unknown motivations.

2.1 Gboard's Architecture and Privacy

Gboard is a virtual keyboard application available for both Android and iOS devices. Importantly, Gboard uses FL to train it's next word prediction model. This is a word level LSTM RNN language model, predicting the probability of the next word given what the user has already typed into the keyboard.

Previous work by Leith *et al.* [11] has shown that it is possible to reconstruct private client data under the honest-but-curious threat model from individual Gboard model updates. Their attack operates by taking advantage of a key

[3] `https://play.google.com/store/apps/details?id=com.google.android.inputmethod.latin`. Last accessed 2023-06-19. 5 billion+ downloads.

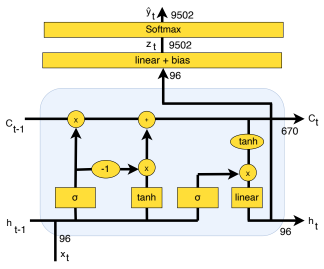


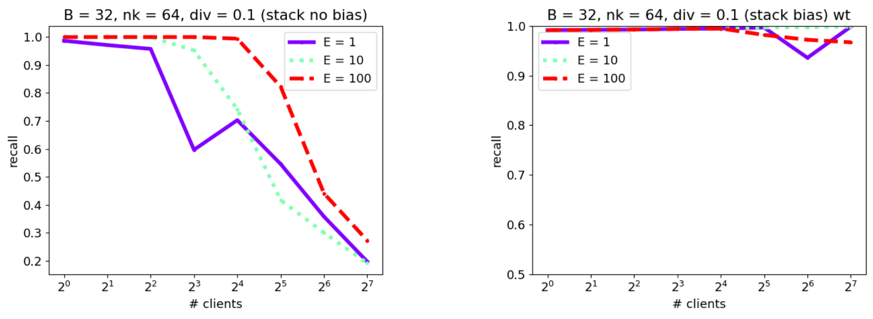
Fig. 1. Gboard LSTM layer takes as input a dense vector $x_t \in \mathbb{R}^D, D = 96$ representing a typed word and outputs a dense vector $h_t \in \mathbb{R}^D$. This output is then mapped to vector $z_t \in \mathbb{R}^V, V = 9502$ (the size of the dictionary) with the value of each element being the raw logit for the corresponding dictionary word. A softmax layer then normalises the raw z_t values to give a vector $\hat{y}_t$ of probabilities.

property of the gradients of the final, fully connected layer. For an example sequence $(\mathbf{x}^{(1)}, c_1), (\mathbf{x}^{(2)}, c_2), ..., (\mathbf{x}^{(T)}, c_T)$ of T total timesteps, where $\mathbf{x}^{(t)} \in \mathbb{R}^D$ is the current word embedding, and c_t is the next word, we have the total loss function $L = \sum_{t=1}^{T} \ell_t(\mathbf{x}^{(t)}, c_t)$, where $\ell_t = -\log \frac{e^{\mathbf{z}^{(t,c_t)}}}{\sum_j^V e^{\mathbf{z}^{(t,j)}}}$, is cross entropy loss at timestep t. The vector $\mathbf{z}^{(t)} = \mathbf{h}^{(t)} W + \mathbf{b}$ is the model's raw logit output at timestep t, a vector of length V. We use the notation $\mathbf{z}^{(t,i)}$ to index the output vector, and $\mathbf{h}^{(t)}$ is the previous layer activation at timestep t. Then, we have the derivative of the loss at timestep t w.r.t the i-th neuron bias $\mathbf{b}_i$, $\frac{\partial L}{\partial \mathbf{b}_i} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{z}^{(t,i)}}$. As shown in [14], the sign of the derivative of the cross entropy loss w.r.t to the outputs is negative only if $i = c_t$ i.e. token i was typed. Thus the index of the negative bias gradients reveal the typed words of the users participating in the given FL round. This information is a privacy breach in and of itself, and can be used to mount further attacks to reconstruct original sentences, exploiting the generative nature of langauge models [6, 11].

This attack exploits two aspects of the architecture, namely (i) the inputs are echoed by the outputs and (ii) there exists a final layer bias whose gradients reveal the typed words. Figure 2 shows how a simple change such as removing the bias can drastically impact the attack's performance. In our experiments[4], each client trains their local LSTM model on sentences taken from the stack exchange data dump [3] following the `FederatedAveraging` algorithm described in [9]. The server, then mounts the attack described above on the difference between the initial shared model and and the final aggregate. This attack can also be performed on the gradients of the final layer weight matrix W, as shown in [14]. However, they consider batch sizes of only 1 example. Crucially, the attack degrades when a greater number of clients is used in SA when no bias is present, however, when a bias is present, SA appears to provide no extra benefit

[4] Our code for these FL experiments is available at `https://anonymous.4open.science/r/fl-attacks-gboard2-89B2/`

in terms of privacy. Ultimately, architectural changes can both enhance and degrade privacy. Gupta *et al.* [6] propose the use of pre trained word embeddings, eliminating the gradients for the final layer. Additionally, removing a bias from the final layer can help in preventing this type of attack.



(a) No bias (b) With bias

Fig. 2. Effect of model architecture on the performance of a word reconstruction attack against Gboard updates (see later for further details). Simple changes like adding a bias to the final layer can subvert the protection afforded by Secure Aggregation. Here, each client trains their local model using 64 sentences, a batch size of 32. We plot the word recall results for a varying number of local epochs .

3 Verifiable FL Implementations

Production implementations of FL algorithms are embedded within larger software systems that include telemetry, remote configuration, device authentication/attestation etc. It is, of course, the privacy of the system as a whole that is of concern to users. We show below that poor implementations can easily allow de-anonymisation of devices and users as well as creating new potential channels for attacks. Our GBoard measurement study also highlights that public documentation and support for independent evaluation of developed apps by the FL community is important both to verify privacy claims and to build confidence in users that apps employing FL are indeed safe to use.

3.1 Vulnerabilities in the Gboard FL Implementation

In the subsequent sections we provide examples of ways one can exploit the telemetry sent by Gboard's FL implementation to de-anonymise users and bypass aggregation. We monitor the traffic sent by an android[5] device, using a man-in-the-middle attack implemented using the `mitmproxy` [2] tool suite.

Telemetry Allows for De-Anonymisation In our experiments, we observe that the `eligibility_eval_checkin_request` messages regularly sent by the Google GBoard and Google Messages apps on the handset to Google server `federatedml-pa.googleapis.com` include Google SafetyNet[6] device attestation data. For example:

```
Headers:
:path: /google.internal.federatedml.v2.FederatedTrainingApi/Session
:authority: federatedml-pa.googleapis.com
content-type: application/grpc
Body:
eligibility_eval_checkin_request {
  attestation_measurement: "CgaqR5AG...AEIgA"
  <...>
}
```

The data sent can readily be used for device fingerprinting e.g., it contains the names of cache folders which are randomly generated and so effectively act as device identifiers. Furthermore, Google apps send telemetry data via the private Clearcut logger API of Google Play Services. to the `play.googleapis.com/log/batch` endpoint. Google does not provide public documentation on the data sent to this server, nor on the data format used. However, recent work has reversed engineered a good deal of the message format [8], and we leverage that work here. Each message sent includes an NID cookie and an x-server-token authentication token (which act as device identifiers), followed by a sequence of telemetry protobufs. In our experiments, we observe that Gboard sends telemetry logging FL operations. For example:

```
POST https://play.googleapis.com/log/batch
Headers:
x-server-token : CAESKQDyiOh8EP...25qEp5
cookie : NID=511=DleS...YCwE
Body:
<...>
androidId: 4468978717649541595
<...>
logSourceName: BRELLA
logEntry:
<...>
timestamp: 1681397351854
inAppTrainingService: "com.google.android.inputmethod.latin"
<...>
runID: 40431939126563851
phaseID: "fake_phase_id"
```

When the aim of DP is to ensure that is difficult to determine whether a user contributed to the data, the possibility de-anonymisation removes this guarantee, making it trivial to establish whether a particular user did or did not contribute to model training. These sort of side channel attacks can erode any privacy guarantees provided by FL hardened with SA and DP.

[5] Hardware and software used: Google Pixel 4a, Google Play Services ver. 22.09.20, Google Gboard ver. 12.4.06, rooted using Magisk. Device Settings: following factory reset, settings are left at their defaults.

[6] See, e.g., `https://developer.android.com/training/safetynet/attestation`

Aggregation Bypass and Population Control In the Google FL protocol, handsets can execute an `eligibility_eval` plan and return the response. This includes an initial checkpoint (model parameter values), a local dataset to use, and a TensorFlow graph the handset should execute to update the checkpoint and generate a response. This response is not aggregated and is sent by the handset to the server in subsequent FL `checkin_request` messages.

In the Google FL protocol, each FL model within a Google app on a handset is assigned to a population. There may be multiple models within the same app, e.g., in GBoard there are next word prediction, speed recognition, and emoji prediction models and each may be assigned to a separate population.

The population of a model within an app is assigned a name, e.g., `lstm_federated_training_population`, and defaults to a null value, e.g., "impossible", but can be updated via the Google Play Services Heterodyne service. The Heterodyne service sends details of installed apps to server `www.googleapis.com/experimentsandconfigs` and the server responds with configuration values. For example:

```
POST https://www.googleapis.com/experimentsandconfigs/v1/getExperimentsAndConfigs
Headers:
x-server-token : CAESOQDyiOh8v...Hp4Q==
authorization : Bearer ya29.m.CvkBAZ8...kS1pjLU53
Body:
<...>
androidId: 4468978717649541595
cookie: "NID=511=DleS...YCwE"
<...>
<details of apps installed and their experiment tokens>
```

It can be seen that this request includes multiple device and user identifiers. The `androidId` itself is enough to link the message to the individual device and user (using the messages sent to `android.googleapis.com/checkin` noted above), but in addition, the authorization header is generated using the user's Google account and so is directly linked to the handset user. This means that the server has to hand information that allows the FL population values it assigns to be based on the individual device and user.

4 Reducing the Need for Trust

An honest FL participant may believe that they have preserved their privacy by using FL, yet their data can be readily recovered and tied to them by the coordinating server. This is a direct result of the inordinate amount of trust clients are asked to place in the FL system. Clients must *trust* that the co-ordinating server is faithfully carrying out the FL protocol, clients must *trust* that the other clients are genuine, clients must *trust* that the cryptography behind the PKI in SA is not compromised, clients must *trust* the model architecture has not been designed maliciously, etc.

When the FL system and the model being trained are both operated by a reputable organisation then perhaps such trust can be justified. However, it requires strong governance and oversight of that organisation and the avoidance of potential conflicts of interest (such as the organisation also being a consumer of user data for analytics, advertising etc). When such a level of trust exists then it also begs the question of why not simply send raw client data to the central server and trust that it is stored ephemerally and only used for the purposes of model training in combination with data from sufficiently many other users i.e., the added privacy value of FL seems rather small.

When multiple parties are involved in the FL system, such as with Federated-Learning-as-a-Service (FLaas) [7], establishing sufficient trust seems much harder. For example, suppose an organisation operates FLaas for mobile apps. Then the models to be trained by FL on private client data may be supplied and used by multiple different app developers with a broad geographic spread and different regulatory regimes. As we already know from mobile app stores, users then have only a limited ability to establish developer bona fides and even powerful gate-keepers such as Google and Apple have difficulty regulating developer behaviour. Hence, even when the FLaas provider is trusted, the overall FLaas system need not be trustworthy.

We note, however, that some degree of trust is likely to be asked of FL users. Trying to ensure user privacy when the FL service is actively malicious is probably a hopeless endeavour – the server may insert synthetic devices, manipulate model weights, architecture, compromise the PKI, and training process actively during training and it seems hard to defend against all of these while still providing a useful FL service. The need is to greatly reduce the level of trust asked of users, and thereby provide a better privacy risk-benefit trade-off to them.

5 Conclusion

FL in its current form requires a large amount of trust in the central server on the part of the participating clients. By studying Google's Gboard FL application, we have demonstrated that certain aspects of FL, such as the choice of the model architecture, and their real-life implementation, require special attention. We have built upon the work showing that model architecture affects privacy by providing a real world, in production, example of architectural modification to aid data reconstruction. Additionally, we have also analyzed Google's FL implementation in Gboard and shown that there exist some privacy concerns in the telemetry. User and device de-anonymisation is possible, data aggregation can be bypassed, and the central server maintains control of the population. We hope that the discussions provided here will motivate further research into FL and its added privacy benefit and serve as a potential future roadmap.

Acknowledgements This work was supported by European Union's Horizon 2020 research and innovation programme under grant number 951911 – AI4Media.

References

1. Boenisch, F., Dziedzic, A., Schuster, R., Shamsabadi, A.S., Shumailov, I., Papernot, N.: Reconstructing individual data points in federated learning hardened with differential privacy and secure aggregation (2023)

2. Cortesi, A., Hils, M., Kriechbaumer, T., contributors: mitmproxy: A free and open source interactive HTTPS proxy (v5.01) (2020), `https://mitmproxy.org/`
3. Exchange, S.: Stack exchange data dump (2023), `https://archive.org/details/stackexchange`
4. Fowl, L.H., Geiping, J., Czaja, W., Goldblum, M., Goldstein, T.: Robbing the fed: Directly obtaining private data in federated learning with modified models. In: International Conference on Learning Representations (2022), `https://openreview.net/forum?id=fwzUgo0FM9v`
5. Geiping, J., Bauermeister, H., Dröge, H., Moeller, M.: Inverting gradients - how easy is it to break privacy in federated learning? In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 16937–16947. Curran Associates, Inc. (2020), `https://proceedings.neurips.cc/paper_files/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf`
6. Gupta, S., Huang, Y., Zhong, Z., Gao, T., Li, K., Chen, D.: Recovering private text in federated learning of language models. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (eds.) Advances in Neural Information Processing Systems (2022), `https://openreview.net/forum?id=dqgzfhHd2-`
7. Kourtellis, N., Katevas, K., Perino, D.: Flaas: Federated learning as a service. In: Proceedings of the 1st Workshop on Distributed Machine Learning. p. 7–13. DistributedML'20, Association for Computing Machinery, New York, NY, USA (2020). `https://doi.org/10.1145/3426745.3431337`, `https://doi.org/10.1145/3426745.3431337`
8. Leith, D.: What Data Do The Google Dialer and Messages Apps On Android Send to Google? In: Proc Securecomm (2022)
9. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
10. Pasquini, D., Francati, D., Ateniese, G.: Eluding secure aggregation in federated learning via model inconsistency. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. p. 2429–2443. CCS '22, Association for Computing Machinery, New York, NY, USA (2022). `https://doi.org/10.1145/3548606.3560557`, `https://doi.org/10.1145/3548606.3560557`
11. Suliman, M., Leith, D.: Two models are better than one: Federated learning is not private for google gboard next word prediction (2022)
12. Truex, S., Liu, L., Chow, K.H., Gursoy, M.E., Wei, W.: Ldp-fed: Federated learning with local differential privacy. In: Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking. p. 61–66. EdgeSys '20, Association for Computing Machinery, New York, NY, USA (2020). `https://doi.org/10.1145/3378679.3394533`, `https://doi.org/10.1145/3378679.3394533`
13. Wei, K., Li, J., Ding, M., Ma, C., Yang, H.H., Farokhi, F., Jin, S., Quek, T.Q.S., Vincent Poor, H.: Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security **15**, 3454–3469 (2020). `https://doi.org/10.1109/TIFS.2020.2988575`
14. Zhao, B., Mopuri, K.R., Bilen, H.: idlg: Improved deep leakage from gradients (2020)
15. Zhao, J.C., Elkordy, A.R., Sharma, A., Ezzeldin, Y.H., Avestimehr, S., Bagchi, S.: The resource problem of using linear layer leakage attack in federated learning. arXiv preprint arXiv:2303.14868 (2023)

16. Zhao, J.C., Sharma, A., Elkordy, A.R., Ezzeldin, Y.H., Avestimehr, S., Bagchi, S.: Secure aggregation in federated learning is not private: Leaking user data at large scale through model modification (2023)
17. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf