Analysis of Secured Federated Learning

1st Aayush Jannumahanti
FK32171
aayushj1@umbc.edu

2nd Aksheetha Muthunooru
ME52139
me52139@umbc.edu

3rd Avni Saxena
UG47736
asaxena4@umbc.edu

4th Shashwat Jain
XB36090
xb36090@umbc.edu

***Abstract*—A novel method for remote training of massive deep neural networks called federated learning (FL) keeps participant data on their own devices and only shares model updates with a central server. However, FL's scattered nature creates further dangers brought on by potentially harmful people. In order to evaluate the trade-offs and robustness of FL, we examine attacks against the FL system and build an image classification model using EMNIST dataset in this article**

I. Introduction

Federated Learning (FL) is a promising data privacy technique, including healthcare and financial services. It has attracted a lot of interest and attention due to its capacity to train machine learning models on decentralized data sources without compromising privacy.

The adoption of FL in the healthcare industry makes it possible for diverse healthcare providers to share private patient information, promoting joint research and bettering patient results. FL complies with adding a layer of data privacy while enabling healthcare organizations to make use of a massive amount of data while ensuring patient privacy.

Similarly, Financial organizations are also acknowledging the value of FL for enhancing their offerings and reducing fraudulent activity. This way they can examine patterns and trends in client behavior using FL without jeopardizing the confidentiality of personal financial information.

Although FL is a step closer to data privacy, there are still a number of problems and challenges with it. As discussed there are many uses of FL where FL is deployed as a privacy technique, therefore is very important to dwell on the question that how vulnerable FL still is. Through the examination of two distinct privacy strategies FL and FL with DP—our study tries to delve deeply into the problem.

Our work makes an effort intends to provide light on the efficiency and limitations of these two privacy strategies by conducting a detailed analysis of them. By adding to the vast going on research on FL by applying various data privacy attacks, this study will help people to understand the advantages and disadvantages of Federated learning as a privacy technique.

II. Methodology and Implementation

A. Federated Learning

Federated learning (FL) is a machine learning setting where many clients collaboratively train a model under the orchestration of a central server while keeping the training data decentralized without sharing raw data. It is motivated by the growing need for privacy and security in the era of big data.

We have implemented Federated learning with the help of an open-source framework called TensorFlow Federated (TFF) is made available by TensorFlow.

Here is a general a brief of how TensorFlow Federated incorporates federated learning:

TensorFlow federated: A federated computation, which is a TensorFlow-like computation that uses federated data, is a concept that Federated presents. You can specify computations that are carried out on portions of data that are dispersed over several machines or devices.

- *Federated Learning with Image Classification*
  With the [2] Federated API provided by TFF module we explored it for Federated Learning with Image Classification introduces the key parts of the Federated Learning (FL) API, and demonstrates how to use TFF to simulate federated learning on federated MNIST-like data.

B. Differential Privacy

Differential Privacy (DP) has attracted a lot of interest in the field of data privacy. By incorporating controlled noise into data analysis procedures, it offers a formal and mathematical definition of privacy guarantees.

In this [3] [6], **the aim is to protect the privacy of client data in federated learning (FL) by applying differential privacy (DP) using the adaptive clipping method**. To achieve this, certain steps are followed. First, the model updates from clients are clipped, which means they are limited to a certain range before being sent to the server. This clipping helps control the influence of individual clients on the aggregated model. Then, to further safeguard privacy, noise is added to the combined updates. This noise is in the form of Gaussian noise, which follows a specific distribution. The amount of noise added and the clipping threshold are dynamically determined based on how sensitive the updates are. This adaptation ensures a balance between privacy protection and the utility, or usefulness, of the learned model for each client.

The [6]'s approach to incorporating differential privacy into the federated learning model is inspired by the techniques described in a research paper called "Differentially Private Learning with Adaptive Clipping" authored by Andrew et al. It builds upon the principles discussed in that paper to achieve privacy guarantees. In the tutorial's example, multiple models are trained using 50 clients per round. To gradually increase the privacy level, the noise level is progressively raised by

adjusting the ”noise-multiplier.” This parameter controls the ratio between the noise's standard deviation and the clipping norm, which determines the range of the noise. Since adaptive clipping is employed, the actual intensity of the noise varies from round to round, allowing for better privacy protection throughout the training process.

C. *Federated Learning with Differential Privacy*

Federated learning is a step closer to data privacy but is still susceptible to various attacks. To address such issues, federated learning (FL) with differential privacy (DP) combines two potent methodologies.

The ”Differential Privacy in TFF” tutorial offers a comprehensive guide on implementing differential privacy techniques in federated learning using TensorFlow Federated (TFF). It outlines the essential steps required to achieve differential privacy in the federated learning process. The referenced tutorial utilizes Keras, a high-level deep learning API, to define and compile the model architecture for classifying handwritten characters. Also, the tutorial uses stochastic gradient descent (SGD) as an optimizer that is used during the model compilation process in Keras. The SGD optimizer helps update the model parameters based on gradients computed from the training data, contributing to the model's learning process.

In [3], it begins by loading the Federated EMNIST dataset, specially designed for federated learning scenarios where data is distributed across multiple clients. It then proceeds to define the model architecture and sparse categorical cross-entropy loss function responsible for classifying handwritten characters. To ensure privacy, a DP query is constructed using the TFF dp-query module, which introduces privacy-preserving noise to the aggregated model updates. By integrating the DP query with the federated averaging process, the tutorial ensures that client updates are aggregated in a differentially private manner. The configuration of privacy parameters, such as epsilon (ϵ), allows users to control the desired level of privacy. Finally, the trained differentially private model's accuracy in character classification and the achieved level of privacy protection are evaluated.

D. *Dataset*

The Federated EMNIST dataset utilized in the [2] [3] is an extended version of the original MNIST dataset, containing handwritten character images. Unlike MNIST, which solely focuses on handwritten digits, EMNIST incorporates a more diverse range of characters, encompassing both digits (0-9) and alphabets (A-Z, lowercase and uppercase). This expansion provides a wider array of characters for classification tasks. Specifically tailored for federated learning, the Federated EMNIST dataset comprises a varied set of clients, with each client possessing a unique subset of EMNIST characters. These clients actively participate in the federated learning system by contributing their local data, facilitating the collective training of a global model while upholding the privacy of their individual data.

By representing a real-world scenario where data is distributed across numerous devices or entities, the Federated EMNIST dataset allows for comprehensive experimentation with federated learning algorithms. It effectively addresses the challenges associated with privacy and decentralized data.

E. *Data privacy attacks*

There are several security and privacy risks associated with federate learning. Malicious local workers can contaminate the model by destroying the integrity, confidentiality, and availability of data before it is trained. Hence to overcome the above mentioned issues, we are testing the system against various attacks.

These attacks are explained as follows -

- **Label Flipping Attack** – A label flipping attack is a type of poisoning attack where an adversary intentionally mislabels their training data, hoping to degrade the performance of the global model. The behavior and decision-making of trained models may be strategically manipulated by adversaries, leading to data breaches or skewed results.
  In [2] [3] applies a label flipping attack to a Federated Learning model using TensorFlow Federated (TFF) and the EMNIST dataset. Each function in the methodology has a unique purpose in generating and deploying the label flipping attack.
  The Function for Flipping Labels: The Function for Flipping Labels: The crux of the label flipping attack is contained in the flip-labels function. This function is designed to modify the labels of the data points in the dataset at random intervals. It generates a new label, and depending on the flip-probability, might substitute this newly generated label for the original one. The likelihood of label alteration is determined by the flip-probability parameter; a flip-probability of 0.3 suggests a 30% probability for a label to be changed.
  Incorporating Label Flipping into the Dataset: The element-n and preprocess-train-dataset functions are responsible for applying label flipping to the complete training dataset. element-fn applies the flip-labels function to each data point, while preprocess-train-dataset uses element-fn on every data point in the dataset, effectively scanning the entire dataset and flipping labels based on the given probability. Subsequently, the dataset undergoes shuffling, repeating, and batching processes.
  Handling the Test Dataset: Handling the Test Dataset: In the case of the test dataset, label flipping is also applied through the preprocess-test-dataset function, but with a flip-probability of 0.0. This means no labels in the test dataset will be changed. This is done because label flipping attacks are usually aimed at the training process, while the testing process remains unaffected to gauge the model's accuracy.
  Loading and Preprocessing Data: The function get-emnist-dataset oversees the loading and preprocessing of the EMNIST dataset. It applies the label flipping attack to

the training dataset with a flip-probability of 0.3, which means that 30% of data points will have their labels replaced by a random label. The test dataset, however, is left unchanged.

This is how label flipping attacks is implemented on a Federated Learning model within the TensorFlow Federated framework. To implement a label flipping attack, you would need to modify the client data during preprocessing. Here's how you could do this: Identify which client(s) will be the adversary. You can choose this randomly, or based on some other criteria. During preprocessing, when the labels are being assigned, flip the labels for the adversary client(s). For a binary classification task, you can just invert the label (i.e., change 0s to 1s and vice versa). For a multi-class classification task like image classification, you can change each label to some other class.

- **Backdoor attacks** –
  A backdoor attack in Federated Learning (FL) involves a malicious client device installs secret access points or 'backdoors' in a federated learning model, which can manipulate the global model's performance. In a federated learning scenario, a backdoor attack could be implemented by having one or more malicious clients modifying their local data in a specific way.

  This section describes the backdoor attack in a federated learning context with the help of [2] [3], specifically leveraging TensorFlow Federated and the EMNIST dataset. The methodology is divided into three key components: the introduction of a backdoor trigger, the modification of preprocessing functions, and the setup of the federated learning environment. **Establishing the Backdoor Trigger -** A function add-trigger-to-image was created to insert a backdoor trigger into an image which is a white pixel, at the top-left corner of the image.

  Adjusting Preprocessing Functions: The preprocess-with-possible-backdoor function was created to integrate and add the backdoor trigger to change the label to a predefined target if the add-backdoor flag is activated. This alters the label to the predefined backdoor target label which is 9 in our case.

  Simulating the Federated Learning Environment: Federated learning scenario was simulated with multiple client datasets, one of which is assumed to be a malicious server. During data preprocessing, the backdoor trigger is added only to this adversarial client's data.

  The methodology effectively introduces a backdoor attack in the federated learning setting. The trained model, when presented with an image containing the backdoor trigger, classifies it as label 9, illustrating the potency of the attack.
- **Model Inversion** – We explored on gradient-based model inversion attack with the help of [1], which allows the malicious server of Federated Learning to reconstruct the private local dataset via shared gradients.

  These methods reconstruct the private images by minimizing the distance between the fake gradients and the received gradients. Each method has its own strategy, such as the distance metric and regularization terms.

  This attack tries to reconstruct the private training data by optimizing the fake data to generate gradients close enough to the received gradients from the client.

  Exchanging gradients is a widely used in modern multi-node machine learning system (e.g., distributed training, collaborative learning). For a long time, people believed that gradients are safe to share: i.e., the training data will not be leaked by gradients exchange. However it is possible to obtain the private training data from the publicly shared gradients. We name this leakage as Deep Leakage from Gradient and empirically validate the effectiveness on both computer vision and natural language processing tasks. Experimental results show that attack is much stronger than previous approaches: the recovery is pixel-wise accurate for images and token-wise matching for texts. Thereby we want to raise people's awareness to rethink the gradient's safety.

  DLG (Deep Leakage from Gradients) and iDLG (Improved Deep Leakage from Gradients) are two gradient-based model inversion attack strategies utilized in the code. Here's a comparison between DLG and iDLG:

  DLG Attack:

  DLG is a fundamental gradient-based model inversion attack tool. It uses the gradients of the target model to reconstruct the inputs or produce adversarial examples. The DLG attack attempts to deduce sensitive information from the target model by exploiting information leakage from gradients. During the federated learning process, gradients from the client are collected and used to rebuild inputs or produce adversarial instances. DLG does not use explicit regularization strategies to improve the quality of adversarial examples [1].

  iDLG Attack:

  iDLG is an upgraded version of the DLG attack that addresses its shortcomings while increasing its efficacy. To increase the quality of adversarial samples, iDLG integrates extra regularization algorithms. The iDLG attack contains L2 regularization in the code, which improves the fidelity and usability of the generated adversarial samples. The regularization coefficients in iDLG are calibrated to strike a compromise between creating realistic examples and retaining the attack's effectiveness. iDLG intends to decrease defects and improve the visual quality of adversarial samples by using regularization [1].

  Comparative Evaluation:

  In the context of federated learning, DLG and iDLG both use gradient information for model inversion attacks. DLG is a fundamental attack technique, but iDLG is an enhanced variant with additional regularization approaches. With the addition of regularization, iDLG aims to provide higher-quality adversarial instances than DLG. Regularization approaches in iDLG helps in the reduction of defects and the improvement of visual fidelity of

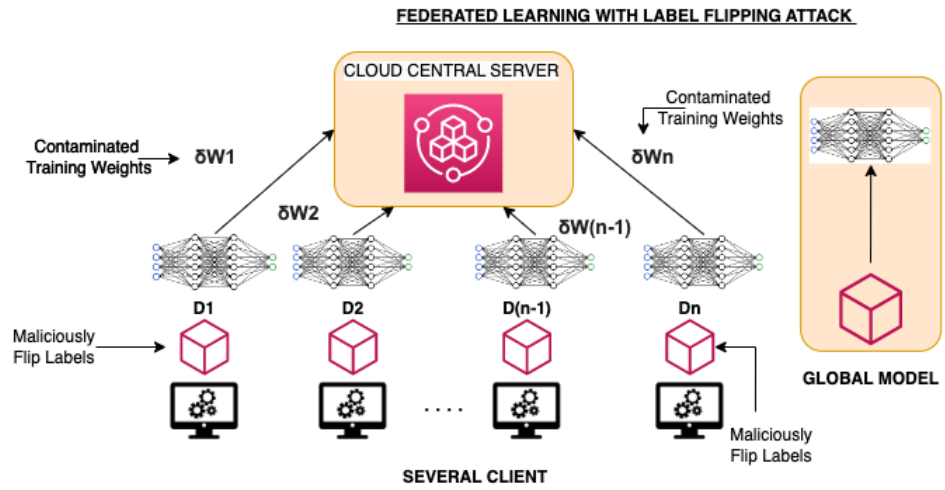

Fig. 1. Architecture of Federated Learning + Differential privacy with Label Flipping Attack Implementation

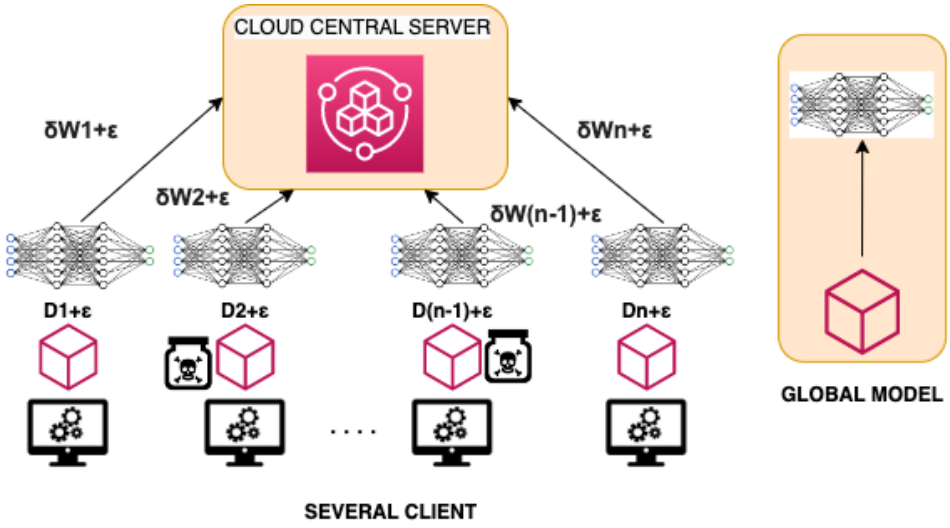

Fig. 2. Architecture of Federated Learning + Differential privacy with Backdoor attack

generated adversarial samples. Both attacks take use of the possible privacy and security vulnerabilities involved with federated learning gradient sharing. The decision between DLG and iDLG is determined by the attack's specific requirements and aims, with iDLG providing a more refined and enhanced approach.

In general, iDLG is a step up from the standard DLG attack since it uses regularization techniques to raise the quality and effectiveness of the adversarial samples produced during the model inversion procedure.

III. Related Work

For the advancement in distributed machine learning where data privacy is also maintained, McMahan et al. (2017) initially presented federated learning (FL) "**Communication-Efficient Learning of Deep Networks from Decentralized Data**" [5]. Since the server (aggregator) never receives the training data, FL is in favor of machine learning with privacy and regulatory restrictions. The tests we conducted in synchronous update rounds using typical FL settings are discussed and analyzed in this publication.

In our project, we drew inspiration from the work of Andrew et al. in their paper titled "**Differentially Private Learning with Adaptive Clipping** [6]." Their work offered a comprehensive understanding of adaptive clipping, which is a crucial component in preserving privacy while maintaining utility in machine learning models. By studying and applying their methodology, we were able to effectively understanding differential privacy and adapt it to the specific requirements of our project. The contribution of Andrew et al. played a pivotal role in our study, enabling us to explore and evaluate the effectiveness of differential privacy in the context of our project's objectives.

In the field of FL research, various attacks and defense

mechanisms have been explored. One notable work in this area is the research conducted by Abdur R. Shahid, titled "**Label Flipping Data Poisoning Attack Against Wearable Human Activity Recognition System**" [4]. This study focuses on investigating a specific type of attack known as data poisoning, which targets wearable human activity recognition systems. While Shahid's work provides valuable insights into the label flipping attack in this context, our study takes a broader perspective by examining the concept of secured federated learning and federated learning with differential privacy.

IV. RESULTS

In this section of this project report, we did a thorough analysis of outcomes attained, concentrating on the two most important aspects: loss and privacy. We study the results of Federated Learning (FL) and Federated Learning with Differential Privacy (FL-DP) with various variations on model performance and privacy protection. We also thoroughly assess the impact on FL and FL-DP of three different attacks. We hope to provide valuable insights into the trade-offs between model accuracy and privacy preservation.

A. Federated Learning v/s Federated Learning with differential Privacy

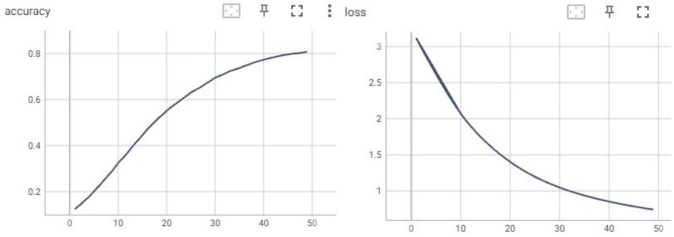

Fig. 3. Accuracy and Loss graphs for FL-image classification

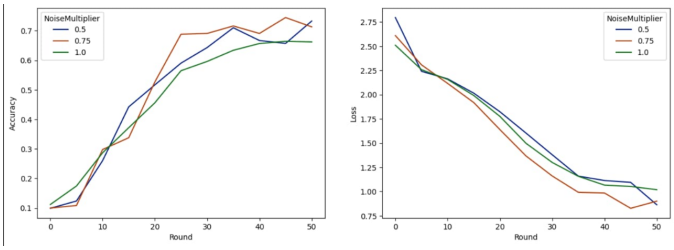

Fig. 4. Accuracy and Loss graphs for FL with DP

The Figures 3 and 4 shows the loss and accuracy graph for FL model vs FL with DP model (containing various noise factor values of 0.5, 0.75 and 1.0) respectively.

The loss graph of any federated learning model typically shows a decreasing trend, as the model becomes more accurate with additional rounds of training. Similarly, federated learning with differential privacy also aims to achieve the same goal while also ensuring that the client's private data remains protected. Differential privacy involves adding noise to the model updates. This added noise can make the model updates a bit less accurate, resulting in a higher loss but still overall shows similar results while also adding another layer of privacy using DP.

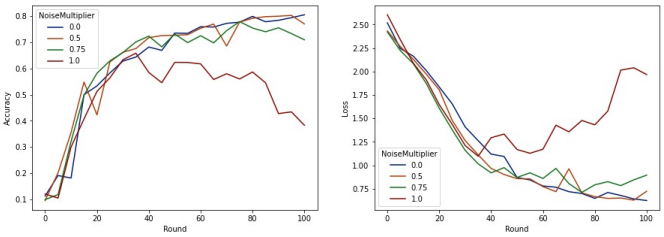

Fig. 5. Accuracy and Loss graphs for FL with DP for 100 rounds

Also, as it be seen from Figures 4 and 5, we have analyzed FL with DP technique for different values of noise, to see how much utility is decreasing with different levels of noise. As can be seen from Figure 5, if we increase the number of rounds of training to 100 with a high value of noise factor, after round 50, the overall accuracy of the graph starts deteriorating to a very high degree. Therefore, we need to settle on a certain value of noise where we can get optimal value of privacy vs utility.

B. Label Flipping Attack- FL v/s FL with DP

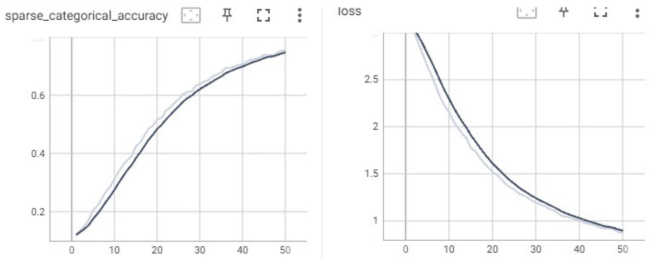

Fig. 6. Accuracy and Loss graphs for FL with Label Flipping attack

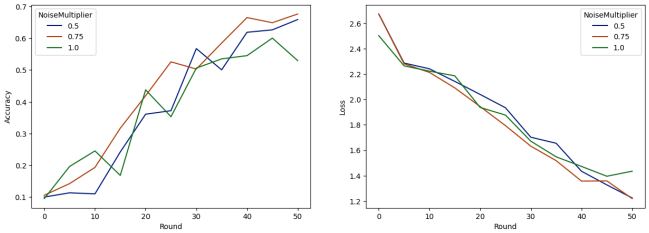

Fig. 7. Accuracy and Loss graphs for FL+DP with Label Flipping attack (Label Probability Factor=0.3)

To understand the Label Flipping attack, we compare the label flipping attack on a FL model with a FL model with DP technique implemented on it. As can be seen from Figure 6 - for the Label flipping attack on FL model with no differential privacy, the loss value is equivalent to 0.9 at the convergence of the model. Now, comparing the same with Figure 7, we

can see that the loss value at convergence is 1.1 which is again almost equivalent. So, as can be seen, in case of Figure 7, even after adding noise to the data (because differential privacy is used here) to increase the user level privacy, we are achieving comparable levels of performance in both the scenarios. Looking at figure 8, after further increasing the level

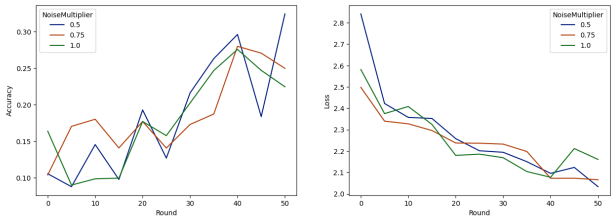

Fig. 8. Accuracy and Loss graphs for FL+DP with Label Flipping attack (Label Probability Factor=0.7)

of attack on the model by increasing the value of label flipping factor to 0.7, the loss value does not converge as it doing in the case of Figure 7 (FL with DP with Label Flipping Factor 0.3). Hence, if the level of attack to the model is increased, the model performance decreases. This can be seen from the Figure 8 where the loss value is equivalent to 2.1 instead of 1.1 for Figure 7 at convergence of the model.

Overall, performing Label Flipping attack on a differtial privacy FL model helps gauranteeing user privacy to a certain extent but it can still be susceptible to the affects of the attack if we increase the level of attack factor by tweaking by value of label flipping factor.

C. *Backdoor attack-FL v/s FL with DP*

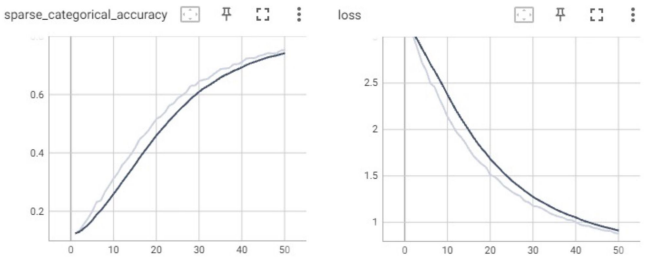

Fig. 9. Accuracy and Loss graphs for FL with BackDoor Attack

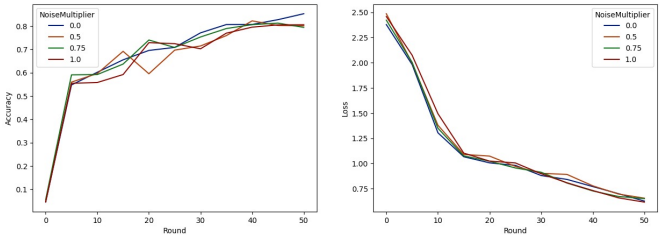

Fig. 10. Accuracy and Loss graphs for FL+DP with BackDoor Attack

We have also implemented backdoor attack on FL model and FL model with DP. This can be seen in Figure 9 and Figure 10. Figure 9 showcases the results of Backdoor attack implemented on an FL model and Figure 10 showcases the results of Backdoor attack implemented on FL model with DP. Now, as can be seen from Figure 9 and Figure 10, in both cases, the model starts converging after round 50. But after careful examination, Figure 10 can be seen having a steeper curve where the loss value of the curve drops significantly even in the initial few rounds. This is not the same in case of Figure 9, where the loss value is decreasing gradually. Here, the 'loss' value is a metric that is used to measure how well the model's predictions are and how well it is performing overall. Therefore, examing the results shows that the FL with DP model is performing much better meanwhile also ensuring the user level privacy as integrating DP is adding an extra layer of privacy for the data.

D. *Model Inversion*

Fig. 11. Reconstructed single data which we recovered from the received gradients with batch size 1.

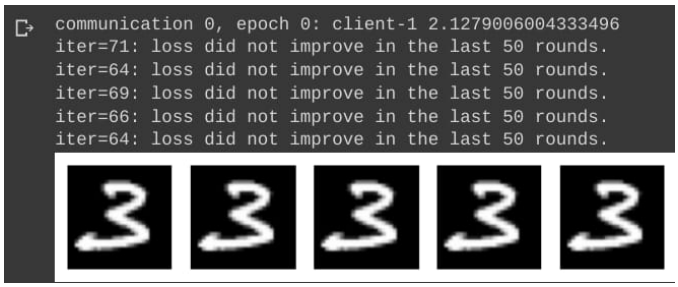

Fig. 12. Deep leaked gradient (DLG) attack with distance metric L2 norm and optimized labels

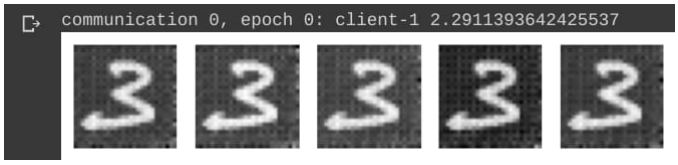

Fig. 13. Improved Deep leaked gradient (iDLG) which has distance metric is L2 norm which analytically estimates a label from the gradients

The reconstruction of a single data point is shown in Fig. 11. With a batch size of 1, it restores the original data by

Fig. 14. Reconstructed batched data simulated with the larger batch size and recovered three images

Fig. 15. Gradient model inversion attack for the large batched data with distance metric L2 norm

utilizing the obtained gradients. The process of reconstruction demonstrates the effectiveness of the suggested method for precisely reconstructing individual data points.

The Deep Leaked Gradient (DLG) attack carried out via a malicious federated server is shown in Fig. 12. The attack uses optimized labels and an L2 norm distance metric. The graphic highlights the possible threat posed by a malicious server having access to the gradients by visualizing the effects of the DLG attack on the system. The integrity of the system may be compromised by the GradientInversionAttackServerManager, who is in charge of coordinating the DLG attack, and manipulating the gradients to enhance the predicted labels.

The Improved Deep Leaked Gradient (iDLG) attack is described in Fig. 13. In this attack, labels are predicted from gradients using analytical estimate and an L2 norm distance metric. The accuracy rates of the DLG attack (client-2.1799) and the iDLG attack (client-2.2911) are compared in the figure. It demonstrates that the iDLG attack has a greater accuracy rate, demonstrating the efficiency of its gradient manipulation to enhance anticipated labels.

Focusing on Fig. 14, it shows how the data has been simulated with a greater batch size. The goal is to extract three particular images from the batch of data. The successful reconstruction of these photos is shown in the figure, illustrating how the suggested methodology may retrieve several data points at once.

Finally, Fig 15 presents the Gradient Model Inversion Attack for large batched data. This attack employs an L2 norm distance metric, analytically estimates labels from the gradients, and incorporates regularization techniques such as total-variance, L2, Bn, and group-consistency. The figure visually illustrates that the generated fake gradients closely resemble the received gradients, indicating the success of the attack in compromising the system's integrity.

In summary, these figures provide a detailed analysis and visualization of the different attacks and reconstruction techniques employed in the study. They highlight the potential vulnerabilities of the system to malicious attacks and the effectiveness of the proposed methodologies in manipulating gradients and reconstructing data points.

V. Limitation

Our current implementation focuses only on Label flipping and Backdoor attacks which are mainly data poisoning attacks on federated learning using TFF. These attacks mainly focus on manipulating the data at the preprocessing stage in the federated learning model. Incorporating other types of attacks such as membership inference, eavesdropping attacks, and model inversion attacks using TFF library will likely be challenging and may not work well for complex models and datasets like the one used in the project since federated learning and federated learning with DP in itself is very secure data privacy approach. Therefore, for this study, we mainly focused on applying attacks on the data preprocessing stage. Thus more research can be done in this direction.

AIJack primarily concentrates on attacks using gradients, such DLG and iDLG. While these algorithms may be helpful in some circumstances, they might not cover all potential attack techniques. For the implementation of various sorts of attacks, such as optimization-based or black-box attacks, users may need to extend AIJack or rely on other frameworks. Because AIJack was created exclusively for PyTorch, it might not operate perfectly with other deep learning frameworks like TensorFlow or Keras. This limits its applicability. Research is ongoing, and new users who are unfamiliar with federated learning may find it difficult to implement attacks against it.

VI. Future Scope

Future plans for our project include several crucial objectives. First and foremost, we will attempt to implement additional data privacy attacks, such as reconstruction and membership inference attacks on federated learning with differential privacy and analysis part of it, and implement the attacks on various datasets with various parameters and gradients. We will then compare the models and attacks and examine how they can be made secure and resistant to those attacks. Fair comparisons between various algorithms and strategies can be facilitated by establishing standardized standards, protocols, and assessment measures for federated learning. This can encourage teamwork, increase reproducibility, and hasten the adoption of federated learning in many fields.

VII. Conclusion

In this project, the concept of federated learning (FL) has been investigated as a method to maintain data privacy while training machine learning models on decentralized data sources. FL allows participants to keep their data on their own devices and share only model updates with a central server, thus avoiding the need for data centralization. The trade-offs and robustness of FL have been evaluated by examining

various attacks against the FL system and developing an image classification model using the EMNIST dataset. Several data privacy attacks, including label flipping, backdoor, and model inversion attacks, were explored and implemented on FL models. The effectiveness of federated learning with differential privacy (FL-DP) in mitigating these attacks and preserving user privacy was also explored. In result, we looked into accuracy versus round graphs to associate the utility of the frameworks and loss graphs for the performance. The results demonstrated that FL-DP can provide a certain level of privacy protection while maintaining comparable model performance, the unavoidable tradeoff of utility versus privacy. The addition of noise in FL-DP models was found to enhance privacy but slightly impact model accuracy. Additionally, FL-DP models were generally more resilient to attacks compared to FL models without differential privacy. However, it is important to acknowledge that this study primarily focused on data poisoning attacks during the data preprocessing stage, and other types of attacks, such as membership inference and eavesdropping attacks, were not extensively covered. In conclusion, federated learning holds promise as a technique for maintaining data privacy, allowing collaboration and utilization of decentralized data while preserving individual privacy. By understanding the vulnerabilities and limitations of FL and exploring techniques like FL-DP, the security and privacy of machine learning models in decentralized environments can be enhanced. Further research and advancements are necessary to address other types of attacks and improve the overall security of federated learning systems.

VIII. Individual Contribution

A. *Aayush*

In my role, I have gained extensive knowledge and practical experience in Privacy Enhancing techniques focused on user-centric data privacy. Throughout my work, I have diligently explored multiple frameworks, libraries, and APIs associated with Federated Learning, aiming to grasp the technical intricacies of the project while prioritizing the pressing need for user-centric data privacy. To implement the concept of federated computation, I successfully incorporated Tensorflow Federated and Tensorflow Privacy libraries into the project. Based on valuable feedback received from Dr. Yus, I also devoted my efforts to leveraging OpenMined's PySyft library. I delved deep into the analysis of attacks using the Tff library by Tensorflow. By utilizing the MNIST dataset, I conducted a series of experiments involving various attacks, including the Label Flipping attack, Backdoor attack, and Vanilla Federated Learning. These investigations have yielded fruitful results, providing valuable insights into the security aspects of user-centric data privacy. I firmly believe that our study and its outcomes lay a strong foundation for further advancements in the field. By continuing this research, we can propel ourselves forward and embrace a more secure approach to user-centric data privacy.

B. *Aksheetha Muthunooru*

In my role, I extensively reviewed numerous papers and periodicals on federated learning, immersing myself in the subject matter to gain a comprehensive understanding. I made efforts to implement federated learning packages in Tensorflow, relying on pre-existing tutorials for tasks such as image classification and text generation. However, I encountered several challenges during the implementation process, hindering our progress. We explored the Pysyft framework as a potential solution for implementing data privacy attacks, but regrettably, we faced difficulties in its utilization. Consequently, we made a strategic decision to pivot towards basic federated learning with differential privacy using Tensorflow and PyTorch, focusing on the EMNIST dataset.

To ensure the security and robustness of our federated learning system, I delved into extensive research on various federated learning attacks. These included data poisoning, model inversion, label flipping, backdoor attacks, and potential model applications. By thoroughly investigating these attack vectors, I aimed to identify vulnerabilities and explore potential countermeasures to protect our models and data. Furthermore, I took charge of completing the model training process, meticulously prepared datasets, and conducted comprehensive research on the identified attacks.

Through my contributions, which encompassed literature review, implementation troubleshooting, and in-depth research on federated learning attacks, I played a crucial role in advancing our project. By completing the model training, datasets, and attacks research, I made significant strides in enhancing our understanding of federated learning and fortifying our system against potential threats.

C. *Avni Saxena*

In the course of this project, I did anextensive research endeavor was undertaken to explore the potential of Federated Learning (FL) and related strategies in enhancing data privacy. The project involved a thorough study research papers that focused on Federated Learning with Differential Privacy. with the objective to identify and incorporate research findings that align with the goals and requirements of our project. Consequently, I implemented practical demonstrations of data privacy attacks, including Label Flipping and Backdoor attacks, to expose potential risks during data preprocessing in Federated Learning models. The detailed documentation of these implementations contributes to a robust understanding of the methods used and their impact on model performance and privacy preservation.

Furthermore, I did an exploration extended beyond the specific FL implementation discussed earlier. We delved into additional libraries utilized for Federated Learning, such as Pysyft and Aijack. These libraries were thoroughly studied and evaluated to understand their capabilities, features, and potential applicability in our project.

D. *Shashwat Jain*

I read numerous tutorials and research articles to understand about the procedures involved in federated learning implementation, and I then worked with my team to develop a comprehensive plan. We initially tried to think out a way to implement FL using Tensorflow and GRU, but it wasn't feasible for our project. Along with my teammates, I've been reading a lot of blogs and academic papers about attacks, working with Tensorflow and Pytorch to create attacks such as Label Flipping, Backdoor attacks etc to do a comparative analysis. To achieve the same, I explored multiple frameworks, libraries, and APIs associated with Federated Learning. Apart from the attack mentioned in the project, I also worked on exploring and encorporating various other attacks and in understanding the challenges and the issues faced regarding the same. I really want to thank Dr. Yus for giving me this opportunity to work on this project and his guidance. In our study, we also mention the additional data privacy attacks for which we were unable to conduct a comparative analysis of in terms of attack results. We were able to detect a few attacks with federated learning with differential privacy and model attacks and I am planning to explore the same beyond the scope of this project.

IX. GitHub Project Link

https://github.com/aayushkumarjvs/Analysis-Of-Secured-Federated-Learning

References

[1] KoukyoSyumei, "AIJack: Artificial Intelligence Research Journal Club," [Online]. Available: https://koukyosyumei.github.io/AIJack/index.html.

[2] TensorFlow, "Federated Learning for Image Classification," TensorFlow Tutorials, [Online]. Available: https://www.tensorflow.org/federated/tutorials/federated_learning_for_image_classification.

[3] TensorFlow, "Federated Learning with Differential Privacy," TensorFlow Tutorials, [Online]. Available: https://www.tensorflow.org/federated/tutorials/federated_learning_with_differential_privacy.

[4] "Label Flipping Data Poisoning Attack Against Wearable Human Activity Recognition System" https://arxiv.org/pdf/2208.08433.pdf.

[5] "Communication-Efficient Learning of Deep Networks from Decentralized Data " https://arxiv.org/pdf/1602.05629.pdf.

[6] "Differentially Private Learning with Adaptive Clipping" https://arxiv.org/pdf/1905.03871.pdf.

[7] "Backdoor Attacks and Defenses in Federated Learning: Survey, Challenges and Future Research Directions" https://arxiv.org/pdf/2303.02213.pdf.

[8] "MPAF: Model Poisoning Attacks to Federated Learning based on Fake Clients" https://arxiv.org/pdf/2203.08669.pdf.

[9] "Defending against the Label-flipping Attack in Federated Learning" https://arxiv.org/pdf/2207.01982.pdf.

[10] "Tensorflow Official Documentation by Google" https://www.tensorflow.org/.

[11] "Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives" https://cybersecurity.springeropen.com/articles/10.1186/s42400-021-00105-6.

[12] "A Survey on Security and Privacy Issues in Edge Computing-Assisted Internet of Things" https://arxiv.org/ftp/arxiv/papers/2008/2008.03252.pdf.

[13] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," IEEE Internet of Things Journal, vol. 8, 2021, pp. 5476–5497.

[14] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Madry, B. Li, and T. Goldstein, "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses," IEEE Trans. Pattern Anal. Mach. Intell. PP, 2023.

[15] X. Xu, J. Wu, M. Yang, T. Luo, X. Duan, W. Li, Y. Wu, B. Wu, "Information leakage by model weights on federated learning," Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice.

[16] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," In AISTATS, 2017.

[17] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov, "How to backdoor federated learning," In AISTATS, 2020.

[18] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li, "DBA: Distributed backdoor attacks against federated learning," In ICLR, 2020.

[19] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," In 2018 IEEE Symposium on Security and Privacy (SP), IEEE.

[20] Shenghui Li, Edith Ngai, Fanghua Ye, and Thiemo Voigt, "Auto-weighted Robust Federated Learning with Corrupted Data Sources," arXiv preprint arXiv:2101.05880 (2021).

[21] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Madry, B. Li, and T. Goldstein, "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.

[22] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in Asian Conference on Machine Learning, PMLR, 2011, pp. 97–112.

[23] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label flips attack on support vector machines," in ECAI 2012, IOS Press, 2012, pp. 870–875.

[24] A. Paudice, L. Munoz-Gonzalez, and E. C. Lupu, "Label sanitization against label flipping poisoning attacks," in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2018, pp. 5–15.

[25] N. Baracaldo, B. Chen, H. Ludwig, A. Safavi, and R. Zhang, "Detecting poisoning attacks on machine learning in IoT environments," in 2018 IEEE International Congress on Internet of Things (ICIOT), IEEE, 2018, pp. 57–64.

[26] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth, "Practical secure aggregation for federated learning on user-held data," CoRR, abs/1611.04482, 2016.

[27] Wei, Wenqi, et al. "A framework for evaluating gradient leakage attacks in federated learning," arXiv preprint arXiv:2004.10397 (2020).

[28] Geiping, Jonas, et al. "Inverting gradients-how easy is it to break privacy in federated learning?," Advances in Neural Information Processing Systems 33 (2020): 16937-16947.

Apart from the above reference, a few more references have been included in the relevant sections above that we have looked into and will be using for our further implementation.