TrustNetFL: Enhancing Federated Learning with Trusted Client Aggregation for Improved Security

David Francis Chen*
dvdchen@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Kehan Wang*
wang5221@purdue.edu
Purdue University
West Lafayette, Indiana, USA

Agnideven Palanisamy Sundar
agpalan@iu.edu
Indiana University - Purdue University Indianapolis
Indianapolis, Indiana, USA

Feng Li
fengli@iupui.edu
Indiana University - Purdue University Indianapolis
Indianapolis, Indiana, USA

ABSTRACT

Federated Learning (FL) has emerged as a promising approach for training machine learning models across individual devices while preserving data privacy. However, FL faces many challenges, specifically a vulnerability to adversarial attacks due to its strict adherence to ensuring individual client model and data privacy. To mitigate these issues, dynamic clipping techniques have been proposed which dynamically adjust the gradient clipping threshold during model aggregation. However, current iterations depend on specific and often intensive calculations to determine a clipping threshold which can lead to an over fitting to a specific dataset or attacker model. In this paper, we focus on improving the limitations of existing FL and dynamic clipping approaches by introducing a novel method that incorporates a group of trusted users during the aggregation of client models for a global update. By identifying and utilizing a network of trusted users, our defense method **TrustNetFL** enhances the robustness of model aggregation against malicious updates. This method not only maintains the model's performance but also improves its resistance to adversarial influences. We demonstrate the effectiveness of our defense through extensive experiments thus showcasing its superiority and simplicity in achieving enhanced model security in FL settings.

KEYWORDS

federated learning, backdoor attacks, dynamic clipping, static clipping, noising

ACM Reference Format:
David Francis Chen, Kehan Wang, Agnideven Palanisamy Sundar, and Feng Li. 2023. TrustNetFL: Enhancing Federated Learning with Trusted Client Aggregation for Improved Security. In *Proceedings of the 24th International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (ACM MobiHoc '23).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*Both authors contributed equally to this research.

ACM MobiHoc '23, October 23-26, 2023, Washington D.C.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Federated Learning (FL), an innovative paradigm of machine learning, has emerged as a cornerstone solution for collaborative model training while preserving the privacy of user data [9]. This approach is particularly relevant with the current landscape of data protection regulations as it allows various entities to engage in collective learning without compromising the data confidentiality of their individual and often, unique users. Contrasting to traditional machine learning where an individual client is only able to train on their individual user data, FL's framework empowers the collaboration of independent groups to glean valuable insights from their respective data sources that previously would have been impossible and illegal. This collaborative effort fosters informed decision-making and innovative strategies for all participants to benefit from such as spam email detection for corporate use [12] and training language models for Google [11].

Despite the merits of collaboration, FL faces challenges such as controlling client models and mitigating vulnerabilities introduced by adversarial attacks. Notably, FL's decentralized nature and privacy preserving foundation make it difficult to directly review individual client training as a central aggregator/server. This design characteristic renders FL susceptible to poisoning attacks from individual malicious clients [7]. For example, the absence and/or weaknesses of centralized data filtering mechanisms can expose the global model to biases/backdoors originating from a malicious client's individual dataset [4]. Specific to security concerns, the emergence of backdoor attacks, including pixel pattern-based methods, poses additional risks [3]. These attacks compromise of inserting a designated pixel pattern onto images during the training phase of an individual client model so that it misclassifies the images with the pattern backdoor. When this model is selected and aggregated to the global model, the global model will also misclassify any image it comes across with the same pattern backdoor.

This paper proposes a novel approach, named **TrustNetFL**, by strategically integrating a subset of trusted users during the client model aggregation process. This approach aims to mitigate vulnerabilities and fortify the security posture of FL environments, including defense against backdoor attacks. By harnessing the contributions of trustworthy clients, TrustNetFL seeks to bolster the

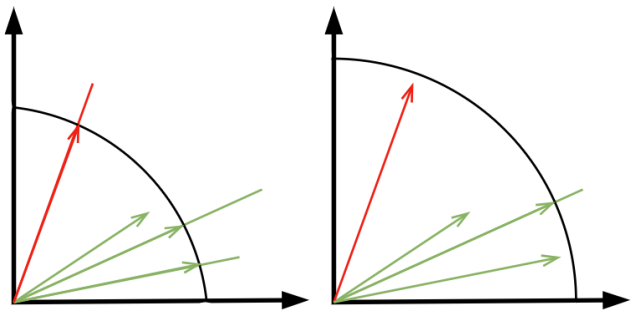
integrity of aggregated models and enhance resistance against adversarial influences, specifically pixel pattern-based attacks. This contribution extends FL's capabilities, offering an innovative solution to its inherent limitations and emerging security challenges, thus making it well-suited for privacy-conscious collaborative learning scenarios.

2 BACKGROUND ON FL DEFENSES

Differential Privacy (DP) is a solution in addressing privacy concerns in model training. It trains models without ever exposing the individual data from users. The process involves privacy-preserving mechanisms such as noise injection and data clipping that are used by aggregators. These techniques not only mask the client models, safeguarding individual data contributors [2, 15], but also act as a strong defense against potential malicious backdoor adversaries attempting to insert backdoors into the central model [10].

2.0.1 Clipping: DP can employ a technique known as "difference clipping" to control the amount of information shared during model updates. This process involves limiting the magnitude of individual model parameter changes before their aggregation [6]. By introducing noise to the updates within a bounded range, differential privacy prevents malicious actors from extracting sensitive information about individual data contributors [2]. Difference clipping strikes a balance between accurate model convergence and protecting the privacy of participants, thus enabling FL to harness the power of diverse data sources while upholding stringent privacy standards.

Difference clipping is typically divided into two categories: static and dynamic [10]. While both have the same goal of being privacy preserving while maintaining overall task accuracy, the parameters used for them are created in two different distinct approaches.



(a) Over Clipping **(b) Under Clipping**

Figure 1: Drawbacks of Static Clipping - Green lines represent benign updates while the red is a malicious backdoor update. The circle represents the preset clipping parameter. The arrow on the lines represents where the model update "magnitude" is clipped.

- Static Clipping: This method requires for a human (often the aggregator) to set a fixed predetermined value to clip by as to limit the size of the individual client updates. Although it is simple and evenly clips all updates to the same "length", this also becomes its Achilles Heel as the model converges and model updates shrink [5, 10]. It's difficult to achieve a balance of clipping and preserving the model's effectiveness as one paper [5] has shown. We also demonstrate this finding in **Fig. 4**. Over clipping (having a small parameter preset shown by **Fig. 1a**) will hinder the contributions of benign clients and the model fails to converge (reach a high accuracy rate). Conversely, under clipping (having a large parameter preset shown by **Fig. 1b**) results in a failure to do its purpose of reducing the effectiveness of backdoors (backdoor effectiveness stays high).
- Dynamic Clipping: This method, on the other hand, adjusts the clipping threshold based on the data or size of the updates. This more logical approach takes into account the data characteristics within each training round, ensuring that larger updates (start of training) receive a more relaxed clipping threshold, while smaller updates (end of training when the model converges) are more tightly clipped [1, 5]. Dynamic clipping can lead to a better convergence performance and utilizes the data more effectively, enhancing the overall performance of the FL process [10, 13, 15]. We will also go over current iterations of and past research on dynamic clipping in Section 2.1.

2.0.2 Noising: In FL, noise is often introduced during the aggregation of model updates from different participants. This prevents an attacker from discerning specific information about any single participant's data by analyzing the aggregated updates. Noise can be added using techniques like noise injection or noise perturbation, which blur the actual update values without significantly affecting the overall learning process [8, 10, 13]. By adding noise, the central aggregator hopes to strike a balance between respecting client privacy while still retaining the ability to extract valuable insights from the privatized client models.

2.1 Related Works

Some works on improving existing dynamic clipping defenses attempt a multitude of methods such as clipping to the mean or median of previous client update L1/L2 Norms [1], a percentile of users [2, 6, 14], or by some multiple coefficient of a mean, median, or calculated value [5, 10, 13, 15].

While all these methods produced positive results in limiting backdoor effects, it should be noted that many of them depended on some arbitrary and often complicated calculation to determine the clipping factor. For example, a scaling factor of 1.1 times the mean [13] or to the 35^{th} percentile [6]. While the introduction of scaling factors improves their performance, there must be careful consideration and statistical analysis to prove that the increased complexity upholds the integrity of the experiment's findings. More paramount is that with the variety of real world applications and model interactions, it would be best for federated defenses to be simple and not depend on an additional confounding variable for its success.

3 TrustNetFL

In FL, the central aggregator (e.g. Google) will most likely also create its own trusted network of client models in addition to the third party models it will receive. Thus, **TrustNetFL** is based on the assumption that the central aggregator's models will always be benign and will always give accurate update values to apply

to the central model. Based on this assumption, TrustNetFL will purposely select from the trusted network each training round and use a mean of all L2-Norm update values given by t number of trusted users (1-5) as the clipping parameter for the other clients selected.

In our experiment, there are 100 clients that will each train an individual model. In each round of training, the central aggregator selects 10 users to combine their models for an update to the central model. From these 100 clients, we always set aside 5 as the trusted users group that can never be selected as backdoor clients. Then from the remaining 95, we select 10 malicious users to insert our pixel pattern backdoor. Finally, within the selection portion, we can choose between 1 to 5 users from the trusted network to be selected within the 10 users with the other 9 to 5 respectively being randomly selected from our pool of 95 users that have "unknown" identities from the central aggregator's perspective (85 good, 10 malicious).

To our knowledge, this is a novel method and in the following sections we will implement it and showcase its viability in protecting FL models from backdoor attacks while not impacting the model's performance.

4 METHODS

For our research, we used a modified version of the GitHub repository that Eugune Bagdasaryan graciously provided with his publication on "How to Backdoor Federated Learning" [3]. Our research is focused specifically on the image classification task using the CIFAR-10 dataset with the pixel pattern attack. We will not be using noising in our experiments to isolate and exclusively determine the effects of clipping. Our end goal is to prove that our defense is able to maintain Main Task Accuracy (**MTA**), while diminishing the Backdoor Accuracy (**BA**).

4.0.1 Setup. We implemented the FL algorithms using TensorFlow on an Anaconda virtual environment running Python 3.7. Experiments were done on two Dell Precision 5470s with an Intel i7-12800H CPU, Nvidia RTX A1000 GPU, 32 GB Ram each on Ubuntu 22.04.

4.0.2 Parameters. In each round, the central aggregator selects 10 clients who are then trained and clipped separately and in sequential order before being aggregated into a global update. The only modifications made to the default settings during our model training were changing the learning rate (0.05), the number of epochs (100), and then for attacks the number of adversaries and their attack weight scale. We adjusted the learning rate and number of epochs to reduce the training time of models so we could quickly prove the effectiveness of **TrustNetFL**.

4.0.3 Procedure. Our research can be categorized into four scenarios that we then add varying additional parameters to show the effectiveness of TrustNetFL. All of these scenarios started with a 10% poisioning rate (10 compromised clients out of 100) and a backdoor weight scale of 2. We chose these values as the poisoning rate is pretty standard within backdoor attack environment and if the scaling rate is too high then it becomes trivial for the defense to clip the longer backdoored models.

(1) **Baseline:** (No Clipping, No Attack) - The first scenario represents the baseline performance, where the model operates without any attacker and without applying clipping. This scenario establishes a reference point for what the normal MTA and BA of the model are without any adversarial influences.
(2) **Attack with No Clipping:** (No Clipping, Has Attack) - The second scenario displays the model's performance when subjected to our backdoor attack without any defense mechanisms such as clipping or noise. This scenario allows us to get MTA and BA results of a backdoored model to prove the potency of the backdoor that we will be defending against. We satisfy the conditions of a successful backdoor as it maintains the MTA and remains stealthy, while achieving a perfect success rate on the backdoor performance, BA.
(3) **Attack with Static Clipping:** (Static Clipping, Has Attack) - The third scenario explores the model's behavior with various parameters of static clipping. As we will show later, utilizing a static parameter is difficult and more importantly, unviable. Finding a balance between maintaining MTA and and reducing BA is extremely difficult and time costly. We will use these results to show the effectiveness and simplicity of TrustNetFL during training.
(4) **Attack with TrustNetFL:** (Dynamic Clipping, Has Attack) - The fourth scenario investigates the model's performance when equipped with our new dynamic clip, TrustNetFL, which is designed to adapt to the converging behavior of our model updates. This dynamic clip aims to provide a more robust and flexible protection against backdoor attacks as we show by outperforming traditional static clipping methods.

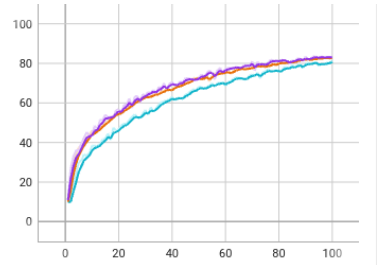
Through these four scenarios, our research aims to offer a comprehensive assessment of TrustNetFL's resilience against backdoor attacks and the impact of static vs. dynamic clipping. The results for all scenarios will be shown on **Figures 2-8**.

5 EXPERIMENT RESULTS

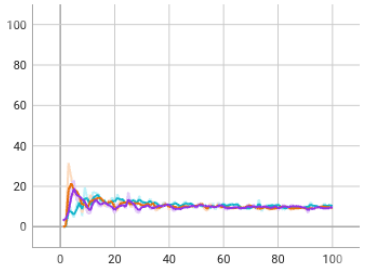
We have taken to liberty to split our Results section into four subsections in the order of our previously mentioned scenarios. All figures (2-8) will have two graphs: the first will display the MTA of our FL model and the second will show the BA. For ease of viewing and comparison, **all our graphs will have axes that go from 0 to 100 by intervals of 20** (x-axis: number of rounds trained, y-axis: accuracy rate as a percentage).

5.0.1 Baseline - (No Clipping, No Attack). In this scenario we ran multiple tests with the parameters mentioned above to get an idea of what results we should be expecting. **Fig. 2** shows the results of 5 of the models trained.

We observe that MTA hovers between 80% and 85% while BA is around 10%. This is expected as the BA is determined by whether the backdoored images are classified correctly into the correct category. Thus, if the backdoor has not been introduced in the round, then the model randomly classifies the backdoored images and because there are 10 categories, it has a one in 10 chance (10% MA) to get it right. Even if the backdoor is only partially introduced, the scaling factor will ensure that the BA will have a large update value to the central model. This process will continue throughout training and the result seen in **Fig. 3b** is that the backdoor has fully been inserted into the



(a) MTA



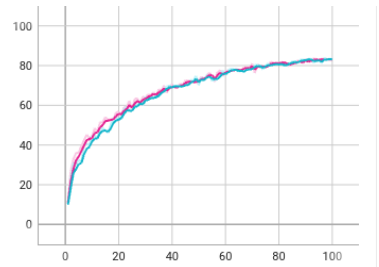
(b) BA

Figure 2: Baseline - The different models/colors don't need to be distinguished as they are trained on identical parameters.

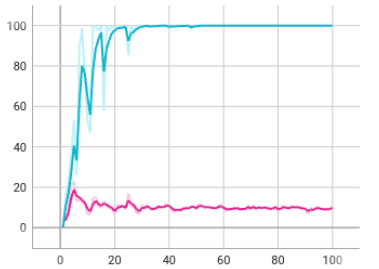
central model which results in the model correctly misclassifying the backdoored images into their respective categories (100% MA).

We will now use this 80% MTA and 10% BA as a comparison to judge the success of our other scenarios.

5.0.2 Attack with No Clipping. From this point on we will show one model for each set of different parameters as to not clutter our graphs and confuse readers. In this test, we simply introduce an attacker to see the effect of a backdoor attack.



(a) MTA

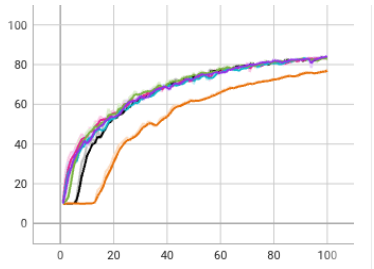


(b) BA

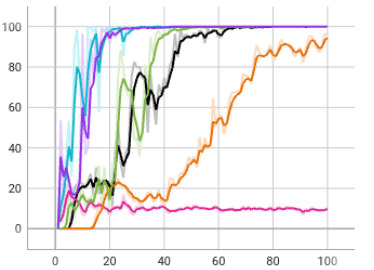
Figure 3: Attack with No Clipping - Pink represents our baseline; Cyan is for our attacker scenario.

In judging the potency of a backdoor, a successful one should maintain the MTA (demonstrating its stealthiness) while also injecting the backdoor into the model. Our attack accomplishes both of those tasks as its MTA has a negligible difference and its BA is 100%. In fact, it was able to achieve that BA of a 100% within the first 20 rounds as shown by **Fig. 3b**.

5.0.3 Attack with Static Clipping. We include the baseline (pink) and attacker (cyan) models in the rest figures (4-8) to act as references for the effectiveness of static clipping. In our testing, we clipped with L-2 Norm values ranging from 5 to 1000. We chose to clip at intervals between 5 to 1000 because those were the size of the updates we saw when printing out the update sizes during training. From 5 to 50 we clipped by intervals of 5 and then after that at 100, 250, 500, and finally 1000. The following figure will explain why we chose such intervals.



(a) MTA



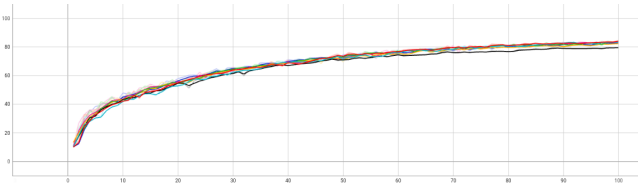
(b) BA

Figure 4: Attack with Static Clipping - Orange is clipping at 5; Black is at 10; Green at 20; Purple at 50

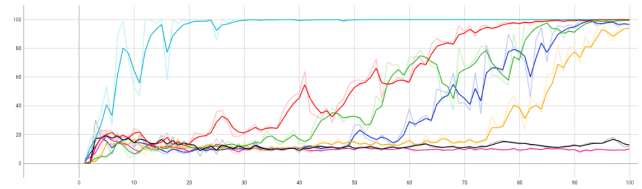
As we observe in **Fig. 4b**, the static clipping value does not affect the overall performance of the backdoor after 100 rounds of training. We have chosen not to include results after the parameter of 50 because as shown by **Fig. 4b**, the performance of our model at 50 (purple) is already virtually the same as having no static parameter (pink) at all. Also as pointed out by others [5], we see that clipping too tightly (parameter of 5) deteriorates the overall performance of the model as it over clips the helpful benign updates (**Fig. 4a**).

5.0.4 Attack with Dynamic Clipping. For **TrustNetFL**, we tried a variety of tests involved with varying the number of trusted users, t, selected during each round ($t/10$). For sake of viewing, we have made these graphs bigger, however we are still presenting all models shown in an identical manner and scale to allow for fair comparison.

There are a few important things to note with TrustNetFL. While using only 1 through 4 trusted users doesn't succeed in reducing



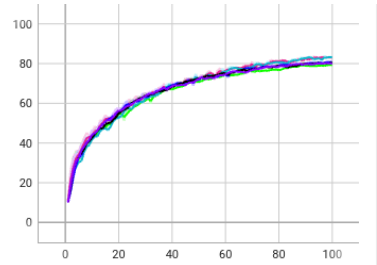
(a) MTA



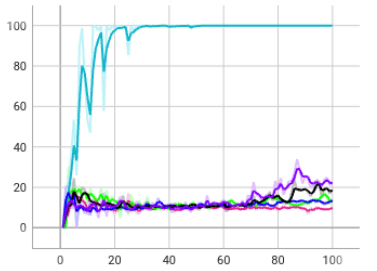
(b) BA

Figure 5: Attack with TrustNetFL - We show our results with a various number, t, of users in our trusted network. Red: t = 1; Green: t = 2; Blue: t = 3; Orange: t = 4; Black: t = 5

the backdoor effectiveness, we see that we initially do mitigate the effect of the backdoor while maintaining MTA (unlike in static clipping). More importantly in **Fig. 5b** we see that using the median of 5 trusted users virtually eliminates the effect of the backdoor.



(a) MTA



(b) BA

Figure 6: TrustNetFL with various backdoor weights - Purple: t = 3 and weight of 5; Black: t = 3 and weight of 10; Blue: t = 4 and weight of 5; Green: t = 4 and weight of 10

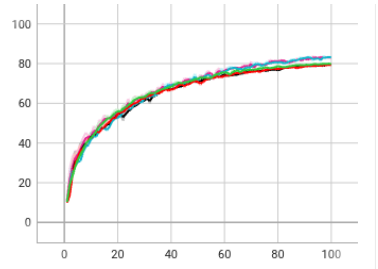
Something else to note from our testing is that the critical value of t to eliminate the backdoor shifted depending on the parameters of backdoor attack. For example, increasing the backdoor weight scale (values of 5 and 10 instead of 2) meant that the backdoor L2-norm updates would be larger, thus making it easier for the median of $t = 3, 4$ trusted users (less representative of all 100 clients as compared to using 5) to clip the backdoor's update size as shown in **Fig. 6**. This also means that if we failed to catch the backdoor that it would proliferate within the central model and quickly increase the BA due to it being a larger update than a backdoor client with a scale of 2. However, we were able to repeatedly get low BA results even with only 3 trusted users (20% was the highest we saw with a weight clip of 5) to calculate a dynamic clipping parameter.

After comparing our results across these four scenarios we are able to conclude that TrustNetFL is viable and is able to achieve significant results in maintaining overall model accuracy while eliminating backdoors.

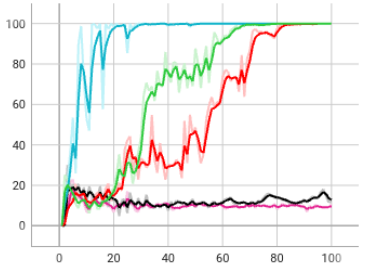
6 LIMITATIONS AND FUTURE WORK

In this section, we want to discuss potential limitations to **TrustNetFL** and make recommendations that could be implemented with future work.

6.0.1 More Backdoor Clients. We initially wanted to test the effectiveness of TrustNetFL if more malicious users (20% and 30% poisoning rate were tested) were included in the 100 clients under the assumption that they could successfully coordinate a pixel pattern attack on our FL model.



(a) MTA

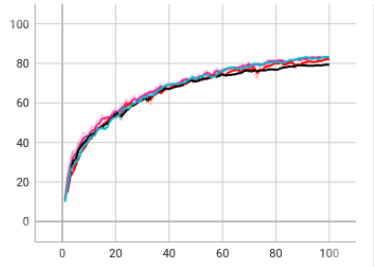


(b) BA

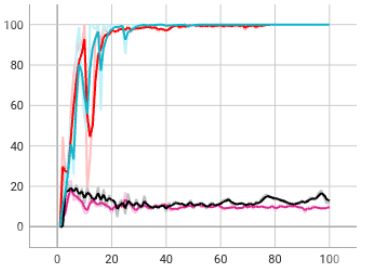
Figure 7: Varying the Attacker Ratio against TrustNetFL - Red is 20 attackers; Green is 30 attackers; Pink is baseline; Cyan is Attacker Only; Black is the model with 5 trusted users

Unfortunately, as shown in **Fig. 7**, TrustNetFL suffers when defending against too many attackers. However, we are assuming that these attackers are all able to sneak into our client pool as well as properly coordinate an attack. We believe that in the real world, this is difficult to accomplish and should not be perceived as a major threat to TrustNetFL.

6.0.2 Malicious Client within Trusted Users. Initially, we also wanted to test to see if TrustNetFL would work if a malicious client was able to sneak into the trusted user network that our central aggregator calculates a dynamic clipping parameter from.



(a) MTA



(b) BA

Figure 8: Introducing a Sneaky Attacker - Red is the malicious client; Pink is baseline; Cyan is Attacker Only; Black is the model with 5 trusted users

Again, unfortunately, as shown in **Fig. 8**, if a malicious client is able to slip into the trusted user pool, it is quite trivial for them to successfully introduce a backdoor. However, we believe that a central aggregator can easily prevent this from happening by screening its trusted users or even by using the client models that it individually trains. If the aggregator is unable to guarantee such terms, then we would advise them to use another defense method as ours depends on the fact that the network users are trustworthy.

With these drawbacks in mind, we want to restate that while TrustNetFL has been shown to be successful, it is not a perfect solution to preventing backdoor attacks in FL. Users should employ the method appropriate to their circumstances as to better their accuracy or chance of success. We would also like to remind readers that malicious users will continue to evolve and create new attackers and as a result we in the cybersecurity community must continue to adapt to these constant advances.

7 CONCLUSION

In this paper, we proposed TrustNetFL for calculating dynamic clip parameters by introducing the concept of trusted users. We presented that by using this trusted pool of users, we can successfully determine a parameter value to dynamically clip the update norms of clients during the aggregation portion of FL. With TrustNetFL, we were able to uphold the overall accuracy of our image classification model while simultaneously reducing the success of the backdoor. While our test environment is not an ideal representation of the variety of real time scenarios that federated learning defenses can be applied, our results provide a clear path for the introduction of trusted users during client model aggregation.

8 ACKNOWLEDGEMENTS

This work was conducted in its entirety through IUPUI's REU Site that has a focus on Mobile Cloud and Data Security. Furthermore, it is financially supported by the National Science Foundation under Grant No. CNS-1852105. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, Oct 2016.

[2] Galen Andrew, Om Thakkar, H. Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping, 2022.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning, 2019.

[4] Kallista A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *CoRR*, abs/1902.01046, 2019.

[5] Jie Fu, Zhili Chen, and Xiao Han. Adap dp-fl: Differentially private federated learning with adaptive noise. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 656–663, 2022.

[6] Yifan Guo, Qianlong Wang, Tianxi Ji, Xufei Wang, and Pan Li. Resisting distributed backdoor attacks in federated learning: A dynamic norm clipping approach. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 1172–1182, 2021.

[7] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista A. Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *CoRR*, abs/1912.04977, 2019.

[8] Haolin Liu, Chenyu Li, Bochao Liu, Pengju Wang, Shiming Ge, and Weiping Wang. Differentially private learning with grouped gradient clipping. In *ACM Multimedia Asia*, MMAsia '21, New York, NY, USA, 2022. Association for Computing Machinery.

[9] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[10] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, Farinaz Koushanfar, Ahmad-Reza Sadeghi, and Thomas Schneider. Flame: Taming backdoors in federated learning, 2022.

[11] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *CoRR*, abs/2103.00710, 2021.

[12] Ikram Ul Haq, Paul Black, Iqbal Gondal, Joarder Kamruzzaman, Paul Watters, and A. S. M. Kayes. Spam email categorization with nlp and using federated deep learning. In *Advanced Data Mining and Applications: 18th International Conference, ADMA 2022, Brisbane, QLD, Australia, November 28–30, 2022, Proceedings, Part II*, page 15–27, Berlin, Heidelberg, 2022. Springer-Verlag.

[13] Koen Lennart van der Veen, Ruben Seggers, Peter Bloem, and Giorgio Patrini. Three tools for practical differential privacy, 2018.

[14] Ning Wang, Yang Xiao, Yimin Chen, Ning Zhang, Wenjing Lou, and Y. Thomas Hou. Squeezing more utility via adaptive clipping on differentially private gradients in federated meta-learning. In *Proceedings of the 38th Annual Computer Security Applications Conference*, ACSAC '22, page 647–657, New York, NY, USA, 2022. Association for Computing Machinery.

[15] Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Jinfeng Yi. Understanding clipping for federated learning: Convergence and client-level differential privacy. *CoRR*, abs/2106.13673, 2021.