BAFFLE: TOWARDS RESOLVING FEDERATED LEARNING'S DILEMMA - THWARTING BACKDOOR AND INFERENCE ATTACKS

ABSTRACT

Recently, federated learning (FL) has been subject to both security and privacy attacks posing a dilemmatic challenge on the underlying algorithmic designs: On the one hand, FL is shown to be vulnerable to backdoor attacks that stealthily manipulate the global model output using malicious model updates, and on the other hand, FL is shown vulnerable to inference attacks by a malicious aggregator inferring information about clients' data from their model updates. Unfortunately, existing defenses against these attacks are insufficient and mitigating both attacks at the same time is highly challenging, because while defeating backdoor attacks requires the analysis of model updates, protection against inference attacks prohibits access to the model updates to avoid information leakage. In this work, we introduce BAFFLE, a novel in-depth defense for FL that tackles this challenge. To mitigate backdoor attacks, it applies a multilayered defense by using a *Model Filtering* layer to detect and reject malicious model updates and a *Poison Elimination* layer to eliminate any effect of a remaining undetected weak manipulation. To impede inference attacks, we build private BAFFLE that securely evaluates the BAFFLE algorithm under encryption using sophisticated secure computation techniques. We extensively evaluate BAFFLE against state-of-the-art backdoor attacks on several datasets and applications, including image classification, word prediction, and IoT intrusion detection. We show that BAFFLE can entirely remove backdoors with a negligible effect on accuracy and that private BAFFLE is practical.

1 INTRODUCTION

Federated learning (FL) is an emerging collaborative machine learning trend with many applications such as next word prediction for mobile keyboards (McMahan & Ramage, 2017), medical imaging (Sheller et al., 2018a), and intrusion detection for IoT (Nguyen et al., 2019). In FL, clients locally train model updates using private data and provide these to a central aggregator who combines them to a *global model* that is sent back to clients for the next training iteration. FL offers efficiency and scalability as the training is distributed among many clients and executed in parallel (Bonawitz et al., 2019). In particular, FL improves privacy by enabling clients to keep their training data locally (McMahan et al., 2017). This is not only relevant for compliance to legal obligations such as the GDPR (2018), but also in general when processing personal and sensitive data.

Despite its benefits, FL is vulnerable to *backdoor* (Bagdasaryan et al., 2020; Nguyen et al., 2020; Xie et al., 2020) and *inference attacks* (Pyrgelis et al., 2018; Shokri et al., 2017; Ganju et al., 2018). In the former, the adversary stealthily manipulates the global model so that attacker-chosen inputs result in wrong predictions chosen by the adversary. Existing backdoor defenses, e.g., (Shen et al., 2016; Blanchard et al., 2017) fail to effectively protect against state-of-the-art backdoor attacks, e.g., constrain-and-scale (Bagdasaryan et al., 2020) and DBA (Xie et al., 2020). In inference attacks, the adversary aims at learning information about the clients' local data by analyzing their model updates. Mitigating both attack types at the same time is highly challenging due to a dilemma: Backdoor defenses require access to the clients' model updates, whereas inference mitigation strategies prohibit this to avoid information leakage. No solution currently exists that defends against both attacks at the same time (§6).

Our Goals and Contributions. In this paper, we provide the following contributions:

1. BAFFLE, a novel privacy-preserving defense approach, that efficiently mitigates backdoor attacks on FL. It is based on a novel adaptive clustering approach (§3.1) for filtering out malicious model updates. In addition, it uses adaptive smoothing (§3.2) to eliminate potentially remaining malicious model contributions. BAFFLE is able to mitigate more complex attack scenarios like the

simultaneous injection of different backdoors by several adversaries that cannot be handled in existing defenses (§3).
2. We design tailored efficient secure (two-party) computation protocols for BAFFLE resulting in private BAFFLE, the *first* privacy-preserving backdoor defense that also inhibits inference attacks (§4). To the best of our knowledge, no existing defense against backdoor attacks preserves the privacy of the clients' data (§6).
3. We demonstrate BAFFLE's effectiveness against backdoor attacks through an extensive evaluation on various datasets and applications (§5). Beyond mitigating state-of-the-art backdoor attacks, we also show that BAFFLE succeeds to thwart adaptive attacks that optimize the attack strategy to circumvent BAFFLE (§5.1).
4. We evaluate the overhead of applying secure two-party computation to demonstrate the efficiency of private BAFFLE. A training iteration of private BAFFLE for a neural network with 2.7 million parameters and 50 clients on CIFAR-10 takes less than 13 minutes (§5.3).

2 BACKGROUND AND PROBLEM SETTING

Federated learning (FL) is a concept for distributed machine learning where K clients and an aggregator A collaboratively build a global model G (McMahan et al., 2017). In training round $t \in [1, T]$, each client $i \in [1, K]$ locally trains a local model W_i (with p parameters/weights $w_i^1, \ldots, w_i^p$) based on the previous global model G_{t-1} using its local data D_i and sends W_i to A. Then, A aggregates the received models W_i into the new global model G_t by averaging the local models (weighted by the number of training samples used to train it): $G_t = \Sigma_{i=1}^{K} \frac{n_i \times W_i}{n}$, where $n_i = \|D_i\|, n = \Sigma_{i=1}^{K} n_i$ (cf. Alg. 2 and Alg. 3 in §A for details). In practice, previous works employ equal weights ($n_i = n/K$) for the contributions of all clients (Bagdasaryan et al., 2020; Xie et al., 2020). We adopt this approach, i.e., we set $G_t = \Sigma_{i=1}^{K} \frac{W_i}{K}$.

Adversary model: In typical FL settings, there are two adversaries: malicious clients that try to inject backdoors into the global model and honest-but-curious (a.k.a. semi-honest) aggregators that correctly compute and follow the training protocols, but aim at (passively) gaining information about the training data of the clients through inference attacks (Bonawitz et al., 2017). The former type of adversary $\mathcal{A}^c$ has full control over K' ($K' < \frac{K}{2}$) clients and their training data, processes, and parameters (Bagdasaryan et al., 2020). $\mathcal{A}^c$ also has full knowledge of the aggregator's operations, including potentially applied backdooring defenses and can arbitrarily adapt its attack strategy at any time during the training like simultaneously injecting none, one, or several backdoors. However, $\mathcal{A}^c$ has no control over any processes executed at the aggregator nor over the honest clients. The second adversary type, the honest-but-curious aggregator $\mathcal{A}^s$, has access to all local model updates W_i, and can thus perform model inference attacks on each local model W_i to extract information about the corresponding participant's data D_i used for training W_i.

Backdoor attacks. The goals of $\mathcal{A}^c$ are two-fold: (1) *Impact*: $\mathcal{A}^c$ aims at manipulating the global model G_t such that the modified model G'_t provides incorrect predictions $G'_t(x) = c' \neq G_t(x), \forall x \in I_{\mathcal{A}^c}$, where $I_{\mathcal{A}^c}$ is a *trigger set* specific adversary-chosen inputs. (2) *Stealthiness*: In addition, $\mathcal{A}^c$ seeks to make poisoned models and benign models indistinguishable to avoid detection. Model G'_t should therefore perform normally on all other inputs that are not in the trigger set, i.e., $G'_t(x) = G_t(x), \forall x \notin I_{\mathcal{A}^c}$, and the dissimilarity (e.g., Euclidean distance) between a poisoned model W' and a benign model W must be smaller than a threshold ε: $\|W' - W\| < \varepsilon$.

Inference Attacks. The honest-but-curious aggregator $\mathcal{A}^s$ attempts to infer sensitive information about clients' data D_i from their model updates W_i (Pyrgelis et al., 2018; Shokri et al., 2017; Ganju et al., 2018; Carlini et al., 2019; Melis et al., 2019) by maximising the information $\phi_i = \text{Infer}(W_i)$ that $\mathcal{A}^s$ gains about the data D_i of client i by inferring from its corresponding model W_i.

3 BACKDOOR-RESILIENT FEDERATED LEARNING

We introduce BAFFLE, a novel defense against backdoor attacks preventing adversary $\mathcal{A}^c$ from achieving attack stealthiness and impact (cf. §2). $\mathcal{A}^c$ can control the attack impact by, e.g., adjusting the poisoned data rate PDR, i.e., the fraction of poisoned data $D_{\mathcal{A}^c}$ in the training data D

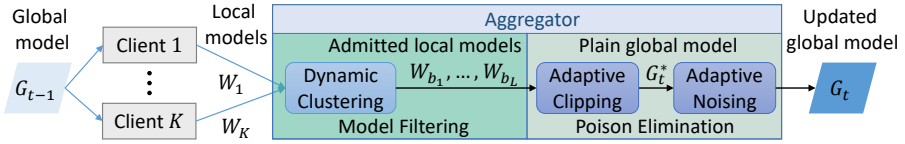


Figure 1: Overview of BAFFLE in round t.

Algorithm 1 BAFFLE

Input: K, G_0, T ▷ K is the number of clients, G_0 is the initial global model, T is the number of training iterations
Output: G_T ▷ G_T is the updated global model after T iterations
for each training iteration t in $[1, T]$ **do**
  for each client i in $[1, K]$ **do**
    $W_i \leftarrow$ CLIENTUPDATE(G_{t-1}) ▷ The aggregator sends G_{t-1} to Client i who trains G_{t-1} using its data D_i locally to achieve local modal W_i and sends W_i back to the aggregator.
  $(c_{11}, \ldots, c_{KK}) \leftarrow$ COSINEDISTANCE$(W_1, \ldots, W_K)$ ▷ $\forall i, j \in (1, \ldots, K)$, c_{ij} is the Cosine distance between W_i and W_j
  $(b_1, \ldots, b_L) \leftarrow$ CLUSTERING$(c_{11}, \ldots, c_{KK})$ ▷ L is the number of admitted models, b_l are the indices of the admitted models
  $(e_1, \ldots, e_K) \leftarrow$ EUCLIDEANDISTISTANCES$(G_{t-1}, (W_1, \ldots, W_K))$ ▷ e_i is the Euclidean distance between G_{t-1} and W_i
  $S_t \leftarrow$ MEDIAN$(e_1, \ldots, e_K)$ ▷ S_t is the adaptive clipping bound at round t
  for each client l in $[1, L]$ **do**
    $W^*_{b_l} \leftarrow W_{b_l} * \text{MIN}(1, S_t/e_{b_l})$ ▷ $W^*_{b_l}$ is the admitted model after clipped by the adaptive clipping bound S_t
  $G^*_t \leftarrow \sum_{l=1}^{L} W^*_{b_l}/L$ ▷ Aggregating, G^*_t is the plain global model before adding noise
  $\sigma \leftarrow \lambda * S_t$ ▷ Adaptive noising level
  $G_t \leftarrow G^*_t + N(0, \sigma)$ ▷ Adaptive noising

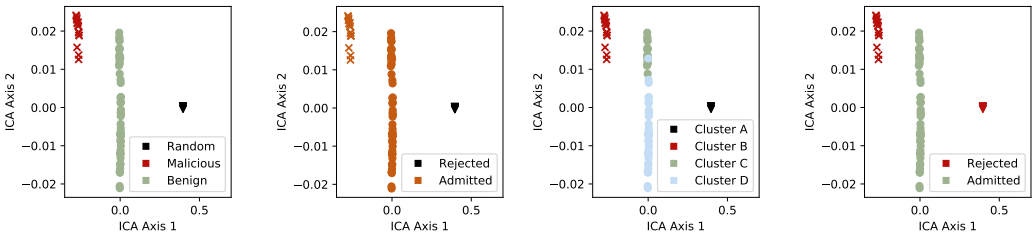
(Eq. 3), or, by tuning the loss-control parameter α that controls the trade-off between backdoor task learning and similarity with the global model (Eq. 4), see §D for details. On one hand, by increasing attack impact, poisoned models become more dissimilar to benign ones, i.e., easier to be detected. One the other hand, if poisoned updates are not well trained on the backdoor to remain undetected, the backdoor can be eliminated more easily. BAFFLE exploits this conflict to realize a multilayer backdoor defense shown in Fig. 1 and Alg. 1. The first layer, called *Model Filtering* (§3.1), uses dynamic clustering to identify and remove potentially poisoned model updates having high attack impact. The second layer, called *Poison Elimination* (§3.2), leverages an adaptive threshold tuning scheme to clip model weights in combination with appropriate noising to smooth out and remove the backdoor impact of potentially surviving poisoned model updates.

3.1 FILTERING POISONED MODELS

The *Model Filtering* layer utilizes a new dynamic clustering approach aiming at excluding models with high attack impact. It overcomes several limitations of existing defenses as (1) it can handle dynamic attack scenarios such as simultaneous injection of multiple backdoors, and (2) it minimizes false positives. Existing defenses (Blanchard et al., 2017; Shen et al., 2016) cluster updates into two groups where the smaller group is always considered potentially malicious and removed, leading to false positives and reduced accuracy when no attack is taking place. More importantly, $\mathcal{A}^c$ may also split compromised clients into several groups injecting different backdoors. A fixed number of clusters bares the risk that poisoned and benign models end up in the same cluster, in particular, if models with different backdoors differ significantly. This is shown in Fig. 2 depicting different clusterings of model updates[1]. Fig. 2a shows the ground truth where $\mathcal{A}^c$ uses two groups of clients: 20 clients inject a backdoor and five provide random models to fool the deployed clustering-based defense. Fig. 2b shows how K-means (as used by Shen et al. (2016)) fails to separate benign and poisoned models so that all poisoned ones end up in the same cluster with the benign models.

Dynamic Clustering. We overcome both challenges by calculating the pairwise Cosine distances measuring the angular differences between all model updates and applying the HDBSCAN clustering algorithm (Campello et al., 2013). The Cosine distance is not affected by attacks that scale updates to boost their impact as this does not change the angle between the updates. While $\mathcal{A}^c$ can easily manipulate the L_2-norms of updates, reducing the Cosine distances decreases the attack impact (Fung et al., 2018). HDBSCAN clusters the models based on their density and dynamically determines the required number of clusters. This can also be a single cluster, preventing false positives in the absence of attacks. Additionally, HDBSCAN labels models as noise if they do not fit

[1]The models were trained for an FL-based Network Intrusion Detection System (NIDS), cf. §E.



(a) Ground truth (b) K-means (c) HDBSCAN (d) BAFFLE

Figure 2: Comparison of clustering quality for (a) ground truth, (b) using K-means with 2 clusters as in Auror (Shen et al., 2016), (c) naively applied HDBSCAN and (d) our approach as in BAFFLE. The models are visualized using Independent Component Analysis (ICA) approach (Jutten & Herault, 1991).

into any cluster. This allows BAFFLE to efficiently handle multiple poisoned models with different backdoors by labeling them as noise to be excluded. We select the minimum cluster size to be at least 50% of the clients, i.e., $\frac{K}{2} + 1$, s.t. it contains the majority of the updates (which we assume to be benign, cf. §2). All remaining (potentially poisoned) models are marked as outliers. This behavior is depicted in Fig. 2d where the two benign clusters C and D from Fig. 2c are merged into one cluster while both malicious and random contributions are labeled as outliers. Hence, to the best of our knowledge, our clustering is the *first* FL backdoor defense for dynamic attacks where the number of injected backdoors varies. The clustering step is shown in Lines 6-7 of Alg. 1 where L models $(W_{b_1}, \ldots, W_{b_L})$ are accepted.

3.2 RESIDUAL POISON ELIMINATION BY SMOOTHING

The *Model Filtering* layer (§3.1) eliminates contributions of poisoned model updates that are not filtered out by adaptive clipping and noising. In contrast to existing defenses that empirically specify a static clipping bound and noise level (and have been shown to be ineffective (Bagdasaryan et al., 2020)), we automatically and adaptively tune these to effectively eliminate backdoors. Our design is also resilient to adversaries that dynamically adapt their attack.

Backdoor embedding makes poisoned models different from benign models. Clipping and noising can be combined to smooth model updates and remove these differences (McMahan et al., 2018). Clipping scales down the model weights to a clipping bound S: $W_i \leftarrow W_i * \text{MIN}(1, S/e_i)$, where e_i is the Euclidean distance (L_2-norm, Def. 1) between W_i and G_{t-1}. Noising refers to a technique that adds noise to a model (controlled by noise level σ): $W^* = W + N(0, \sigma)$, where $N(0, \sigma)$ is a noise generation function, e.g., the Gaussian distribution. While clipping and noising can renove backdoors, previous works (Bagdasaryan et al., 2020) also show that they reduce the global model accuracy on the main task, making it unusable. It is challenging to find an appropriate clipping bound S and a noise level σ that strikes a balance between the accuracy of the main task and effectiveness of the backdoor defense. Both need to be dynamically adapted to model updates in different training iterations and different datasets (§F.1) as well as to dynamic adversaries constantly changing their attack strategy (Bagdasaryan et al., 2020). Note that this use of clipping and noising is different from differential privacy (DP; Dwork & Roth (2014); McMahan et al. (2018)) protecting the confidentiality of clients' data from a curious aggregator and where clients truthfully train their models. In contrast, our scenario concerns malicious clients that intentionally try to backdoor FL. To overcome these challenges, we design our *Poison Elimination* layer for BAFFLE s.t. it automatically determines appropriate values for the clipping bound S and the noise level σ:

Adaptive Clipping. Fig. 3 shows the variation of the average L_2-norms of model updates of benign clients in three different datasets over subsequent training rounds. This shows that the L_2-norms get smaller after each training iteration. To effectively remove backdoors while preserving benign updates unchanged, the clipping bound and noise level must dynamically adapt to this decrease in the L_2-norm. We design an adaptive selection of the clipping threshold S_t for the L_2-norm for each training iteration t. The aggregator selects the median of the L_2-norms of the model updates $(W_1, \ldots, W_K)$ classified as benign in the clustering of our *Model Filtering* layer at iteration t. As we assume that the majority of clients is benign, this ensures that S_t is determined based on a benign model even if some malicious updates were not detected during clustering. We formalize

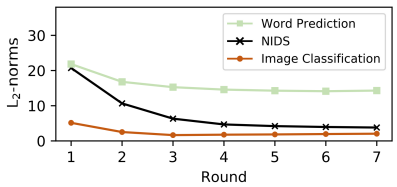


Figure 3: L_2-norms depending on the number of training rounds for different datasets.

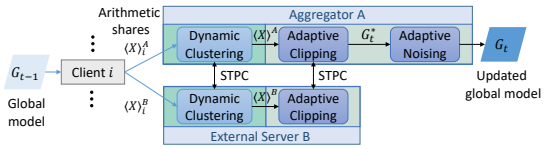


Figure 4: Overview of private BAFFLE in round t using Secure-Two-Party Computation (STPC).

our clipping scheme as follows: $W_{b_l}^* = W_{b_l} * \text{MIN}(1, {}^{S_t}/_{e_{b_l}})$, where $S_t = \text{MEDIAN}(e_1, \ldots, e_L)$ in iteration t, see Lines 8-11 of Alg. 1 for details.

Adaptive noising. We introduce a novel adaptive approach to calculate an appropriate level of noise based on the clipping bound S_t in iteration t. We select the commonly used Gaussian distribution to generate noise that is added to the global model. Let σ be the noise level and let λ be a parameter indicating the product of σ and the clipping bound S_t. Our adaptive noise addition is formalized as follows: $G_t = G_t^* + N(0, \sigma)$, where $\sigma = \lambda S_t$, for a clipping bound S_t and a noise level factor λ, see Lines 13-14 of Alg. 1 for details.

4 PRIVACY-PRESERVING FEDERATED LEARNING

Inference attacks threaten the privacy of FL (cf. §2). They enable the aggregator to infer sensitive information about the clients' training data from the local models. So far, existing defenses against model inference attacks either contradict with backdoor defenses and/or are inefficient (cf. §6). Generally, there are two approaches to protect the privacy of clients' data: differential privacy (DP; Dwork & Roth (2014)) and secure two-party computation (STPC; Yao (1986)). DP is a statistical approach that can be efficiently implemented, but it can only offer high privacy protection at the cost of a significant loss in accuracy due to the noise added to the models (Zhang et al., 2020; Aono et al., 2017; So et al., 2019). In contrast, STPC provides strong privacy guarantees and good efficiency but requires two non-colluding servers. Such servers can, for example, be operated by two competing companies that want to jointly provide a private FL service. STPC allows two parties to securely evaluate a function on their encrypted inputs. Thereby, the parties have only access to so-called secret-shares of the inputs that are completely random and therefore do not leak *any* information besides the final output. The real value can only be obtained if both shares are combined. To provide best efficiency and reasonable security, we chose STPC for BAFFLE. Alternatively, also more parties can be used to achieve better security at the cost of lower efficiency. The technical challenge is to co-design all components of BAFFLE as efficient STPC protocols. For example, we approximate HDBSCAN with the simpler DBSCAN (Ester et al., 1996) to avoid construction of the minimal spanning tree in HDBSCAN as it is very expensive to realize with STPC. Moreover, in STPC, functions to be computed have to be represented as Boolean circuits. For this, we generated several circuit building blocks (e.g., division, square root) using conventional logic synthesis tools similar to (Songhori et al., 2019; Demmler et al., 2015a). Our resulting private BAFFLE is not only the most effective but also the first privacy-preserving backdoor defense for FL.

Private BAFFLE. In private BAFFLE, the distance calculation, clustering, adaptive clipping, and aggregation steps of BAFFLE (cf. Alg. 1) are executed within STPC to protect the privacy of the clients' training data. Our goal is to hide the local models from the aggregator A to prohibit inference attacks on clients' local training data. Fig. 4 shows an overview of private BAFFLE. It involves K clients and two non-colluding servers, called aggregator A and external server B. Each client $i \in \{1, ..., K\}$ splits the parameters of W_i into two Arithmetic shares $\langle X \rangle_i^A$ and $\langle X \rangle_i^B$, such that $W_i = \langle X \rangle_i^A + \langle X \rangle_i^B$ and sends $\langle X \rangle_i^A$ to A and $\langle X \rangle_i^B$ to B. A and B then privately compute the next global model via STPC. We give further details in §C.

5 EVALUATION

We implemented all experiments with the PyTorch framework (pyt, 2019) and used the attack source code provided by Bagdasaryan et al. (2020) and Xie et al. (2020). We reimplemented existing

defenses to compare them with BAFFLE. All experiments that evaluate BAFFLE's effectiveness in defending backdoors were run on a server with 20 Intel Xeon CPU cores, 192 GB RAM, 4 NVIDIA GeForce GPUs (with 11 GB RAM each), and Ubuntu 18.04 LTS OS.

Following previous work on FL and backdooring, we evaluate BAFFLE on three typical applications: word prediction (McMahan & Ramage, 2017) using a LSTM trained on the Reddit dataset (red, 2017), image classification (Bagdasaryan et al., 2020; Xie et al., 2020) using the CIFAR-10 (Krizhevsky & Hinton, 2009), MNIST (LeCun et al., 1998), and Tiny-ImageNet datasets with different architectures, and IoT network intrusion detection (NIDS; Nguyen et al. (2020)). In §E, we detail all datasets used in this work and the experimental setup. To measure the effectiveness of the backdoor attacks and defenses, we consider various metrics: Backdoor Accuracy (BA), Main Task Accuracy (MA), Poisoned Data Rate (PDR), Poisoned Model Rate (PMR), True Positive Rate (TPR), and True Negative Rate (TNR) (all values as percentages) as detailed in §E.2.

5.1 PREVENTING BACKDOOR ATTACKS

Effectiveness of BAFFLE. We evaluate BAFFLE against the state-of-the-art backdoor attacks called constrain-and-scale (Bagdasaryan et al., 2020) and DBA (Bagdasaryan et al., 2020) (cf. §D) using the same attack settings with multiple datasets (cf. Tab. 5 and §E.1). The results are shown in Tab. 1. BAFFLE completely mitigates the constrain-and-scale attack ($BA = 0\%$) for all datasets. The DBA attack is also successfully mitigated ($BA = 3.2\%$, more experiments in §F.8). Moreover, our defense does not affect the main task performance of the system as the Main Task Accuracy (MA) reduces by less than 0.4% in all experiments. BAFFLE is also effective in mitigating state-of-the-art untargeted poisoning attacks (MA increases by 44.59%, more details in §F.4).

We extend our evaluation to various backdoors on three datasets. For NIDS, we evaluate 13 different backdoors and 24 device types (cf. §F.5 and F.5.1), for word prediction 5 different word backdoors (cf. §F.6), and for image classification 90 different image backdoors, which change the output of a whole class to another class (cf. §F.7). In all cases, BAFFLE successfully mitigates the attack while still preserving the MA.

Table 1: Effectiveness of BAFFLE against state-of-the-art attacks for the respective dataset, in terms of Backdoor Accuracy (BA) and Main Task Accuracy (MA).

Attack	Dataset	No Defense		BAFFLE	
		BA	MA	BA	MA
Constrain-and-scale	Reddit	100	22.6	0	22.3
	CIFAR-10	81.9	89.8	0	91.9
	IoT-Traffic	100.0	100.0	0	99.8
DBA	CIFAR-10	93.8	57.4	3.2	76.2
Untargeted Poisoning	CIFAR-10	-	46.72	-	91.31

Comparison to existing defenses. We compare BAFFLE to existing defenses: Krum (Blanchard et al., 2017), FoolsGold (Fung et al., 2018), Auror (Shen et al., 2016), Adaptive Federated Averaging (AFA; Muñoz-González et al. (2019)), and a generalized differential privacy (DP) approach (Bagdasaryan et al., 2020; McMahan et al., 2018). Tab. 2 shows that BAFFLE is effective for all 3 datasets, while previous works fail to mitigate backdoor attacks: BA is mostly negligibly affected. Krum, FoolsGold, Auror, and AFA do not effectively remove poisoned models and BA often remains at 100%. Additionally, the model's MA is negatively impacted. These previously proposed defenses remove many benign updates (cf. §F.1) increasing the PMR and rendering the attack more successful than without these defenses.

Table 2: Effectiveness of BAFFLE in comparison to state-of-the-art defenses for the constrain-and-scale attack on three datasets, in terms of Backdoor Accuracy (BA) and Main Task Accuracy (MA).

Defenses	Reddit		CIFAR-10		IoT-Traffic	
	BA	MA	BA	MA	BA	MA
Benign Setting	-	22.7	-	92.2	-	100.0
No defense	100.0	22.6	81.9	89.8	100.0	100.0
Krum	100.0	9.6	100.0	56.7	100.0	84.0
FoolsGold	**0.0**	**22.5**	100.0	52.3	100.0	99.2
Auror	100.0	**22.5**	100.0	26.1	100.0	96.6
AFA	100.0	22.4	**0.0**	91.7	100.0	87.4
DP	14.0	18.9	**0.0**	78.9	14.8	82.3
BAFFLE	**0.0**	22.3	**0.0**	**91.9**	**0.0**	**99.8**

For example, Reddit's users likely provide different texts such that the distances between benign models are high while the distances between poisoned models are low as they are trained for the

Table 3: Runtime in seconds of standard BAFFLE (S) in comparison to private BAFFLE using STPC (P). K is the number of participating clients. Note that the model size has no effect on the clustering.

K	Cosine Distance						Euclidean Distance + Clipping + Model Aggregation						Clustering	
	Reddit		CIFAR-10		IoT-Traffic		Reddit		CIFAR-10		IoT-Traffic			
	(S)	(P)	(S)	(P)	(S)	(P)	(S)	(P)	(S)	(P)	(S)	(P)	(S)	(P)
10	1.91	297.93	0.05	70.00	0.03	67.67	0.44	218.35	0.27	61.29	0.04	36.85	0.002	3.64
50	50.94	5 259.29	0.80	603.54	0.32	192.47	11.61	594.57	1.82	120.74	0.37	35.04	0.004	41.84
100	213.30	20 560.43	2.66	2 094.51	1.07	554.97	38.82	1 267.35	5.89	219.85	1.03	68.12	0.005	253.87

same backdoor. FoolsGold is only effective on the Reddit dataset ($TPR = 100\%$) because it works well on highly non-independent and identically distributed (non-IID) data (cf. §6). Similarly, AFA only mitigates backdooring on the CIFAR-10 dataset since the data are highly IID (each client is assigned a random set of images) such that the benign models share similar distances to the global model (cf. §6). The differential privacy-based defense is effective, but it significantly reduces MA. For example, it performs best on the CIFAR-10 dataset with $BA = 0$, but MA decreases to 78.9% while BAFFLE increases MA to 91.9% which is close to the benign setting (no attacks), where $MA = 92.2\%$.

Resilience to Adaptive Attacks. Given sufficient knowledge about BAFFLE, an adversary may seek to use adaptive attacks to bypass the defenses. We analyze and evaluate various scenarios and strategies including *changing the injection strategy*, *model alignment*, and *model obfuscation*. Our evaluation results show that BAFFLE is resilient, i.e., mitigates all these attacks effectively (cf. §F.2).

5.2 EFFECTIVENESS OF BAFFLE'S COMPONENTS

Resilience of our in-depth defense approach. To evaluate the effectiveness of our combination of *Model Filtering* and *Poison Elimination*, we conduct experiments in which a sophisticated adversary can freely tune the attack parameter PDR in order to find a setting that evades the filtering layer while still achieving a high BA. We show that the residual poisoned updates are eliminated by *Poison Elimination* in this case. We run experiments covering the full range of PDR values to assess each defense component's effectiveness as well as the complete BAFFLE defense on the IoT-Traffic datasets. The Constrain-and-scale attack is used with the same settings as in §5.1. Fig. 5 shows the BA when using BAFFLE and its individual components depending on the PDR values. As can be seen, *Model Filtering* can reliably identify poisoned models if PDR is above 13%. Below this point, *Model Filtering* becomes ineffective as poisoned models become too indistinguishable from benign ones and cannot be reliably identified. Below this PDR level, however, *Poison Elimination* can effectively remove the impact of poisoned models. Its performance only decreases when PDR is increasing, and the impact of the backdoor functionality is harder to eliminate. However, our BAFFLE effectively combines both defense layers and remains successful for all PDR levels as BA consistently remains close to 0%. A detailed evaluation of the individual components of BAFFLE is given in §F.1.

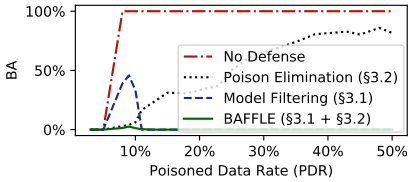


Figure 5: Resilience of each defense layer in comparison to an effective combination in BAFFLE, measured by Backdoor Accuracy (BA).

5.3 PERFORMANCE OF PRIVATE BAFFLE

We evaluate the costs and scalability of BAFFLE when executed in a privacy-preserving manner by varying the number/size of the parameters that affect the three components realized with secure two-party computation (STPC) (cf. §C.1). For our implementation, we use the ABY framework (Demmler et al., 2015b). All STPC results are averaged over 10 experiments and run on two separate servers with Intel Core i9-7960X CPUs with 2.8 GHz and 128 GB RAM connected over a 10 Gbit/s LAN with 0.2 ms RTT.

Tab. 3 shows the runtimes in seconds per training iteration of the Cosine distance, Euclidean distance + clipping + model aggregation, and clustering steps of Alg. 1 in standard (without STPC) and in private BAFFLE (with STPC). The communication costs are given in §F.9. As can be seen, private

BAFFLE causes a significant overhead on the runtime by a factor of up to three orders of magnitude compared to the standard (non-private) BAFFLE. However, even if we consider the largest model (Reddit) with $K = 100$ clients, we have a total server-side runtime of 22 081.65 seconds (≈ 6 hours) for a training iteration with STPC. Such runtime overhead would be acceptable to maintain privacy, especially since mobile phones, which would be a typical type of clients in FL (McMahan et al., 2017), are in any case not always available and connected so that there will be delays in synchronizing clients' model updates in FL. These delays can then also be used to run STPC. Furthermore, achieving provable privacy by using STPC may even motivate more clients to contribute to FL in the first place and provide more data.

Secondly, we measure the effect of approximating HDBSCAN by DBSCAN including the binary search for the neighborhood parameter ϵ (details are given in §C). The results are shown in Tab. 4. As it can be seen, the results are very similar. For some applications, the approximation even performs slightly better than the standard BAFFLE. For example, for CIFAR-10, private BAFFLE correctly filters all poisoned models, while standard BAFFLE accepts a small number ($TNR = 86.2\%$), which is still sufficient to achieve $BA = 0.0\%$. To conclude, private BAFFLE is the first privacy-preserving backdoor defense for FL with significant but manageable overhead and high effectiveness.

Table 4: Effectiveness, in terms of Backdoor Accuracy (BA), Main Task Accuracy (MA), True Positive Rate (TPR), and True Negative Rate (TNR), of standard BAFFLE (S) in comparison to private BAFFLE using STPC (P) in percent.

	Reddit		CIFAR-10		IoT-Traffic	
	(S)	(P)	(S)	(P)	(S)	(P)
BA	0.0	0.0	0.0	0.0	0.0	0.0
MA	22.3	22.2	91.9	91.7	99.8	99.7
TPR	22.2	20.4	23.8	40.8	59.5	51.0
TNR	100.0	100.0	86.2	100.0	100.0	100.0

6 RELATED WORK

Backdoor Defenses. Several backdoor defenses, such as Krum (Blanchard et al., 2017), FoolsGold (Fung et al., 2018), Auror (Shen et al., 2016), and AFA (Muñoz-González et al., 2019), aim at separating benign and malicious model updates. However, they only work under specific assumptions about the underlying data distributions, e.g., Auror and Krum assume that data of benign clients are independent and identically distributed (IID). In contrast, FoolsGold and AFA assume that benign data are non-IID. In addition, FoolsGold assumes that manipulated data is IID. As a result, they are only effective in specific circumstances (cf. §5.1) and cannot handle the simultaneous injection of multiple backdoors (cf. §3.1). In contrast, BAFFLE does not make any assumption about the data distribution (cf. §F.1) and can defend against injection of multiple backdoors (cf. §3.1).

Clipping and noising are known techniques to achieve differential privacy (DP) (Dwork & Roth, 2014; Carlini & Wagner, 2018). However, directly applying these techniques to defend against backdoor attacks is not effective because they significantly decrease the Main Task Accuracy (§5.1). BAFFLE tackles this by (i) identifying and filtering out potential poisoned models that have a high attack impact (cf. §3.1), and (ii) eliminating the residual poison with an appropriate adaptive clipping bound and noise level, such that the Main Task Accuracy is retained (cf. §3.2).

Defenses against Inference Attacks in FL. Bonawitz et al. (2017) use expensive additive masking and secret sharing to hide local updates. Similarly, Chase et al. (2017) train a DNN in a private collaborative fashion by combining multi-party-computation, differential privacy (DP), and secret sharing assuming non-colluding honest-but-curious clients. However, both works are vulnerable to backdoor attacks as they prevent the aggregator from inspecting the model updates. DP (McMahan et al., 2018) limits the success of membership inference attacks that test if a specific data record was used in the training. However, previous works (Melis et al., 2019; Nasr et al., 2019) have shown that this is only successful when thousands of clients are involved or for black-box attacks in which the adversary has no access to model parameters. In private BAFFLE, local model updates are analyzed under encryption, thus the aggregating servers cannot access the updates to run inference attacks while thwarting backdooring.

REFERENCES

Reddit dataset, 2017. https://bigquery.cloud.google.com/dataset/fh-bigquery:reddit_comments.

General Data Protection Regulation, 2018. https://eur-lex.europa.eu/eli/reg/2016/679/oj.

Pytorch, 2019. https://pytorch.org.

Nitin Agrawal, Ali Shahin Shamsabadi, Matt J Kusner, and Adrià Gascón. QUOTIENT: Two-Party Secure Neural Network Training and Prediction. In *Conference on Computer and Communications Security (CCS)*. ACM, 2019.

Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the Mirai Botnet. In *USENIX Security*. Usenix Association, 2017.

Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. Privacy-preserving Deep Learning via Additively Homomorphic Encryption. In *Transactions on Information Forensics and Security (TIFS)*. IEEE, 2017.

Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More Efficient Oblivious Transfer and Extensions for Faster Secure Computation. In *Conference on Computer and Communications Security (CCS)*. ACM, 2013.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How To Backdoor Federated Learning. In *AISTATS*. PMLR, 2020.

Moran Baruch, Gilad Baruch, and Yoav Goldberg. A Little Is Enough: Circumventing Defenses For Distributed Learning. In *NIPS*, 2019.

Donald Beaver. Efficient Multiparty Protocols Using Circuit Randomization. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 1991.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *NIPS*, 2017.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *Conference on Computer and Communications Security (CCS)*. ACM, 2017.

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards Federated Learning at Scale: System Design. In *arXiv preprint:1902.01046*, 2019. URL https://arxiv.org/abs/1902.01046.

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-Based Clustering Based on Hierarchical Density Estimates. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (KDD)*. Springer, 2013.

N. Carlini and D. Wagner. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. In *arXiv preprint:1801.01944*, 2018. URL https://arxiv.org/abs/1801.01944.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. In *USENIX Security*. Usenix Association, 2019.

Melissa Chase, Ran Gilad-Bachrach, Kim Laine, Kristin E Lauter, and Peter Rindal. Private Collaborative Neural Network Learning. In *Cryptology ePrint Archive, Report 2017/762*, 2017. URL https://eprint.iacr.org/2017/762.

Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. Project Adam: Building an Efficient and Scalable Deep Learning Training System. In *USENIX Symposium on Operating Systems Design and Implementation*. Usenix Association, 2014.

Daniel Demmler, Ghada Dessouky, Farinaz Koushanfar, Ahmad-Reza Sadeghi, Thomas Schneider, and Shaza Zeitouni. Automated Synthesis of Optimized Circuits for Secure Computation. In *Conference on Computer and Communications Security (CCS)*. ACM, 2015a.

Daniel Demmler, Thomas Schneider, and Michael Zohner. ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2015b.

Rohan Doshi, Noah Apthorpe, and Nick Feamster. Machine Learning DDoS Detection for Consumer Internet of Things Devices. In *arXiv preprint:1804.04159*, 2018. URL `http://arxiv.org/abs/1804.04159`.

Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. In *Foundations and Trends in Theoretical Computer Science*. now publishers inc., 2014.

Fabienne Eigner, Aniket Kate, Matteo Maffei, Francesca Pampaloni, and Ivan Pryvalov. Differentially Private Data Aggregation with Optimal Utility. In *Annual Computer Security Applications Conference (ACSAC)*. ACM, 2014.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. Springer, 1996.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *USENIX Security*. Usenix Association, 2020.

Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. Mitigating Sybils in Federated Learning Poisoning. In *arXiv preprint:1808.04866*, 2018. URL `http://arxiv.org/abs/1808.04866`.

Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property Inference Attacks on Fully Connected Neural Networks Using Permutation Invariant Representations. In *Conference on Computer and Communications Security (CCS)*. ACM, 2018.

Oded Goldreich, Silvio Micali, and Avi Wigderson. How to Play any Mental Game. In *ACM Symposium on Theory of Computing (STOC)*. ACM, 1987.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016.

Stephen Herwig, Katura Harvey, George Hughey, Richard Roberts, and Dave Levin. Measurement and Analysis of Hajime, a Peer-to-Peer IoT Botnet. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

Russell Impagliazzo and Steven Rudich. Limits on the Provable Consequences of One-way Permutations. In *ACM Symposium on Theory of Computing (STOC)*. ACM, 1989.

Christian Jutten and Jeanny Herault. Blind Separation of Sources, Part I: An Adaptive Algorithm Based on Neuromimetic Architecture. In *Signal Processing*. Elsevier, 1991.

Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A Low Latency Framework for Secure Neural Network Inference. In *USENIX Security*. Usenix Association, 2018.

Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. DDoS in the IoT: Mirai and Other Botnets. In *IEEE Computer*, 2017.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, 2009.

Nishant Kumar, Mayank Rathee, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. CrypTFlow: Secure Tensorflow Inference. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2020.

Peeter Laud. Parallel Oblivious Array Access for Secure Multiparty Computation and Privacy-Preserving Minimum Spanning Trees. In *Privacy Enhancing Technologies Symposium (PETS)*. SCIENDO, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 1998.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training. In *ICLR*, 2018.

Jian Liu, Mika Juuti, Yao Lu, and Nadarajah Asokan. Oblivious Neural Network Predictions via MiniONN Transformations. In *Conference on Computer and Communications Security (CCS)*. ACM, 2017.

Brendan McMahan and Daniel Ramage. Federated learning: Collaborative Machine Learning without Centralized Training Data. In *Google Research Blog*. Google AI, 2017.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*. PMLR, 2017.

H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning Differentially Private Language Models Without Losing Accuracy. In *ICLR*, 2018.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019.

Pratyush Mishra, Ryan Lehmkuhl, Akshayaram Srinivasan, Wenting Zheng, and Raluca Ada Popa. DELPHI: A Cryptographic Inference Service for Neural Networks. In *USENIX Security*. Usenix Association, 2020.

Payman Mohassel and Yupeng Zhang. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017.

Luis Muñoz-González, Kenneth T. Co, and Emil C. Lupu. Byzantine-Robust Federated Machine Learning through Adaptive Model Averaging. In *arXiv preprint:1909.05125*, 2019. URL `http://arxiv.org/abs/1909.05125`.

Moni Naor and Benny Pinkas. Computationally Secure Oblivious Transfer. *Journal of Cryptology*, 2005.

M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019.

Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N. Asokan, and Ahmad-Reza Sadeghi. DÏoT: A Federated Self-learning Anomaly Detection System for IoT. In *International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019.

Thien Duc Nguyen, Phillip Rieger, Markus Miettinen, and Ahmad-Reza Sadeghi. Poisoning Attacks on Federated Learning-Based IoT Intrusion Detection System. In *Workshop on Decentralized IoT Systems and Security @ Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2020.

Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock Knock, Who's There? Membership Inference on Aggregate Location Data. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2018.

Jianji Ren, Haichao Wang, Tingting Hou, Shuai Zheng, and Chaosheng Tang. Federated Learning-Based Computation Offloading Optimization in Edge Computing-Supported Internet of Things. In *IEEE Access*, 2019.

M Sadegh Riazi, Mohammad Samragh, Hao Chen, Kim Laine, Kristin Lauter, and Farinaz Koushanfar. XONN: XNOR-based Oblivious Deep Neural Network Inference. In *USENIX Security*. Usenix Association, 2019.

Sumudu Samarakoon, Mehdi Bennis, Walid Saad, and Merouane Debbah. Federated Learning for Ultra-Reliable Low-Latency V2V Communications. In *Global Communications Conference (GLOBCOM)*. IEEE, 2018.

Joseph Schneible and Alex Lu. Anomaly Detection on the Edge. In *IEEE Military Communications Conference*. IEEE, 2017.

Thomas Schneider and Michael Zohner. GMW vs. Yao? Efficient Secure Two-Party Computation with Low Depth Circuits. In *Financial Crypto*. Springer, 2013.

Micah Sheller, Anthony Reina, Brandon Edwards, Jason Martin, and Spyridon Bakas. Federated Learning for Medical Imaging. In *Intel AI*, 2018a.

Micah Sheller, Anthony Reina, Brandon Edwards, Jason Martin, and Spyridon Bakas. Multi-Institutional Deep Learning Modeling Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation. In *Brain Lesion Workshop*, 2018b.

Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: Defending Against Poisoning Attacks in Collaborative Deep Learning Systems. In *Annual Computer Security Applications Conference (ACSAC)*. ACM, 2016.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017.

Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. In *IEEE Transactions on Mobile Computing (TMC)*. IEEE, 2018.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated Multi-Task Learning. In *NIPS*, 2017.

Jinhyun So, Basak Guler, A. Salman Avestimehr, and Payman Mohassel. CodedPrivateML: A Fast and Privacy-Preserving Framework for Distributed Machine Learning. Cryptology ePrint Archive, Report 2019/140, 2019. URL `https://eprint.iacr.org/2019/140`.

Saleh Soltan, Prateek Mittal, and Vincent Poor. BlackIoT: IoT Botnet of High Wattage Devices Can Disrupt the Power Grid. In *USENIX Security*. Usenix Association, 2018.

Ebrahim M Songhori, Siam U Hussain, Ahmad-Reza Sadeghi, Thomas Schneider, and Farinaz Koushanfar. Tinygarble: Highly Compressed and Scalable Sequential Garbled Circuits. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019.

Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K. Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive Federated Learning in Resource Constrained Edge Computing Systems. In *IEEE Journal on Selected Areas in Communications*, 2019.

Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. DBA: Distributed Backdoor Attacks against Federated Learning. In *ICLR*, 2020.

Andrew Chi-Chih Yao. How to Generate and Exchange Secrets. In *Symposium on Foundations of Computer Science (FOCS)*. IEEE, 1986.

Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *ICML*. PMLR, 2018.

Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning. In *USENIX Security*. Usenix Association, 2020.

A FEDERATED-AVERAGING ALGORITHM

The FedAvg aggregation rule is formalized in Alg. 2. Alg. 3 describes the client part of the training in FL.

Algorithm 2 FedAvg (Aggregator-side execution)

Input: K, G_0, T ▷ K is the number of clients, G_0 is the initial global model, T is the number of training iterations
Output: G_T ▷ G_T is the global model after T iterations
for each training iteration t in $[1, T]$ **do**
  for each client i in $[1, K]$ **do**
    $W_i \leftarrow$ CLIENTUPDATE(G_{t-1}) ▷ The Aggregator sends G_{t-1} to Client i. The client trains G_{t-1} using its data D_i locally to achieve W_i and sends W_i back to the Aggregator.
  $G_t \leftarrow \sum_{i=1}^{K} n_i W_i / n$ ▷ Aggregating

Algorithm 3 LocalTrain

▷ Once Client i receives G_{t-1}, it triggers LOCALTRAIN(G_{t-1}, D_i) using its data D_i and sends W_i back to the Aggregator
function LOCALTRAIN(G_{t-1}, D_i)
  $W_i \leftarrow G_{t-1}$
  for each batch $b \subset D_i$ **do**
    $W_i \leftarrow W_i - \eta \nabla \ell(b, W_i)$ ▷ $\nabla \ell(b, W_k)$ denotes the gradient of the loss function ℓ for a training data batch b and η is the used learning rate
  return W_i

B MODEL SIMILARITY MEASURES

Two measures are commonly used for evaluating the similarity between models: the L_2-norm (Euclidean distance) and the Cosine distance. A model $W = (w^1, w^2, \ldots, w^p)$ consists of p model parameters $w^k, k \in [0, p]$. The similarity measures between two models W_i and W_j, where $0 \leq i, j \leq K$ and K is the number of clients, can therefore be defined as follows:

Definition 1 (L_2-norm Distance)**.** *The L_2-norm distance dl_{ij} between two models W_i and W_j with p parameters, where $0 \leq i, j \leq K$, is the root of the squared parameter differences and is defined as:*

$$dl_{ij} = \|W_i - W_j\| \quad = \sqrt{\sum_{k=1}^{p} (w_i^k - w_j^k)^2}. \tag{1}$$

Definition 2 (Cosine Distance)**.** *The Cosine distance dc_{ij} between two models W_i and W_j with p parameters, where $0 \leq i, j \leq K$, measures the angular difference between the models' parameters and is defined as:*

$$\begin{aligned} dc_{ij} &= 1 - \frac{W_i W_j}{\|W_i\| \, \|W_j\|} \\ &= 1 - \frac{\sum_{k=1}^{p} w_i^k w_j^k}{\sqrt{\sum_{k=1}^{p} {w_i^k}^2} \, \sqrt{\sum_{k=1}^{p} {w_j^k}^2}}. \end{aligned} \tag{2}$$

C DETAILS ON STPC AND PRIVATE BAFFLE

Semi-honest Security. The semi-honest security model is standard in the security and privacy community (Mohassel & Zhang, 2017; Juvekar et al., 2018; Mishra et al., 2020; Liu et al., 2017; Agrawal et al., 2019; Kumar et al., 2020; Riazi et al., 2019) and can be justified by legal regulations such as the GDPR that mandate companies to properly protect users' data. Furthermore, service providers, e.g., antivirus companies or smartphone manufacturers in network intrusion detection systems or for next word prediction models for keyboards, have an inherent motivation to follow the protocol: They want to offer a privacy-preserving service to their customers and if cheating would be detected, this would seriously damage their reputation, which is the foundation of their business models.

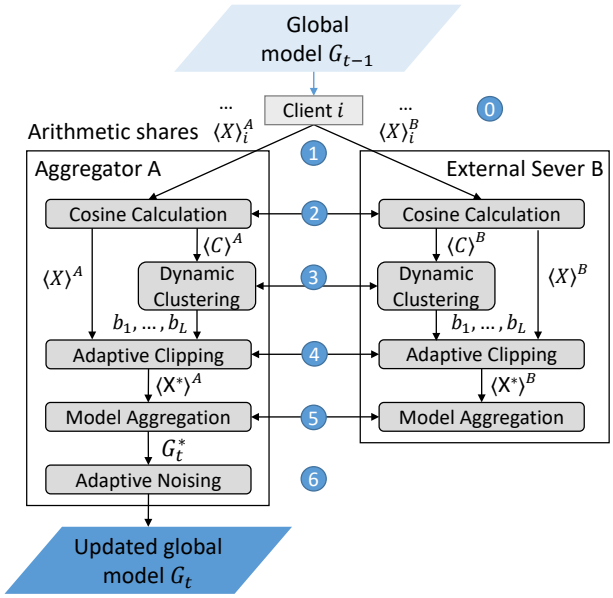


Figure 6: Private BAFFLE processes in round t.

STPC. To design the STPC protocols of BAFFLE, we use a combination of three prominent STPC techniques:Yao's garbled circuits (Yao, 1986) for the secure evaluation of Boolean circuits in a constant number of rounds, as well as Boolean/Arithmetic sharing for the secure evaluation of Boolean/Arithmetic circuits with one round of interaction per layer of AND/Multiplication gates using the protocol of Goldreich-Micali-Wigderson (Goldreich et al., 1987).

Yao's Garbled Circuits (GC). Yao introduced GCs (Yao, 1986) for STPC in 1986. The protocol is run between two parties called *garbler* and *evaluator*. The garbler generates the garbled circuit (GC) corresponding to the Boolean circuit to be evaluated securely by associating two random keys per wire that represent the bit values $\{0, 1\}$. The garbler then sends the GC together with the keys for his inputs to the evaluator. The evaluator obliviously obtains the keys for his inputs via Oblivious Transfer (OT)[2] (Impagliazzo & Rudich (1989); Naor & Pinkas (2005)), and evaluates the circuit to obtain the output key. Finally, the evaluator maps the output key to the real output.

Boolean/Arithmetic Sharing. For every ℓ-bit value v, party P_i for $i \in \{0, 1\}$ holds an additive sharing of the value denoted by $[\mathsf{v}]_i$ such that $\mathsf{v} = [\mathsf{v}]_0 + [\mathsf{v}]_1 \pmod{2^\ell}$. To securely evaluate a multiplication gate, the parties use Beaver's circuit randomization technique (Beaver, 1991) where the additive sharing of a random arithmetic triple is generated in the setup phase (Demmler et al., 2015b). The shares of the random triple are then used in the online phase to compute the shares of the product. In this line of work, the GMW protocol (Goldreich et al., 1987; Asharov et al., 2013; Schneider & Zohner, 2013) takes a function represented as Boolean circuit and the values are secret-shared using XOR-based secret sharing (i.e., $\ell = 1$).

C.1 PRIVATE BAFFLE

Fig. 6 shows the detailed processes of private BAFFLE that is outlined in §4. In 0, each client $i \in [1, K]$ determines its local model in a training round t. In 1, it splits the parameters of W_i into two Arithmetic shares $\langle X\rangle_i^A$ and $\langle X\rangle_i^B$, such that $W_i = \langle X\rangle_i^A + \langle X\rangle_i^B$. The shares are sent to the aggregator A and the external server B over a secure channel.

Let c_{ij} denote the Cosine distance (cf. Eq. 2 in §B) between two models W_i and W_j, where $i, j \in [1, K]$, and let $C = \{c_{11}, \ldots, c_{KK}\}$ be the set of all pairwise distances. In 2, A and B privately

[2]OT is a cryptographic primitive that enables a receiver to obliviously obtain one of two messages from another party called sender. Thereby, the sender learns nothing about which message was chosen by the receiver and the receiver does not learn anything about the message he did not chose.

calculate the set C and receive an arithmetic share of the set's elements as output, i.e., A receives $\langle C \rangle^A = \{\langle c_{11} \rangle^A, \ldots, \langle c_{KK} \rangle^A\}$ and B receives the respective $\langle C \rangle^B$. Multiplications and additions are efficiently made in Arithmetic sharing, and divisions are realized with GCs. A truncation is needed after each multiplication to preserve the size of the fractional part in fixed-point arithmetic. It can be efficiently realized with Boolean sharing, where the least significant bits are cut. This truncation method has on average a minor impact on the accuracy (Mohassel & Zhang, 2017).

Clustering. In 3, clustering is applied to separate benign and malicious models based on similarities between the Cosine distances in C (cf. Line 7 of Alg. 1). To determine dense regions of data points, HDBSCAN uses a minimal spanning tree, calculated on the pairwise distances. As the construction of the minimal spanning tree is expensive to realize with STPC (Laud, 2015), we use as approximation a privacy-preserving version of DBSCAN (Ester et al., 1996), a simplified version of HDBSCAN (Campello et al., 2013) that fixes the neighborhood notion to a maximum distance between two elements by using a parameter called ϵ. The main difference between HDBSCAN and DBSCAN is that DBSCAN cannot handle clusters with varying densities very well, but as we create only a single cluster this is not problematic. We evaluate the accuracy of this approximation in §5.3. To determine an appropriate ϵ-value, we conduct a binary search with several clusterings and varying ϵ-values until one cluster contains exactly $\frac{K}{2} + 1$ elements. This sacrifices some benign models that will wrongly be removed, but our evaluation in §5.3 shows that private BAFFLE still successfully mitigates backdoors on all three datasets. Furthermore, this leaks only two bits of information to the servers, namely, if one cluster has the $\frac{K}{2} + 1$ elements and if the boundary values for ϵ were changed. After determining the right ϵ-value, a final clustering is executed and the resulting cluster indices are opened to A and B to enable them to determine the accepted models in 4. Moreover, A and B can also see who submitted a suspicious model but nothing about this client's training data. DBSCAN's second parameter, called $minPts$ and denoting the minimum cluster size, is set to $\frac{K}{2}$. The clustering outputs a list of clients with accepted models: $N = \{b_1, \ldots, b_L\}$, $L = \frac{K}{2} + 1$. For clustering, we purely rely on GC as it mainly works on binary values.

Euclidean Distance, Clipping, and Model Aggregation. Let e_i, $i \in \{1, \ldots, K\}$, denote the Euclidean distance between a model W_i and the previous global model G_{t-1} and let $E = \{e_1, \ldots, e_K\}$ indicate the set of these distances. In 4, A and B privately calculate E such that A receives $\langle E \rangle^A = \{\langle e_1 \rangle^A, \ldots, \langle e_K \rangle^A\}$ and B receives the respective $\langle E \rangle^B$ as output. There, additions and multiplications are done in Arithmetic sharing, and square roots are calculated with GCs. Afterwards, each model W_i is clipped based on its Euclidean distance e_i to the previous global model G_{t-1}. To clip a model, the calculation of the median of Euclidean distances of the accepted models of the clients in N is done with Boolean sharing and the division and the minimum determination are done with GCs. Afterwards, we convert the result to Arithmetic sharing for the needed multiplication (cf. Line 11 of Alg. 1). In 5, the clipped and accepted models are aggregated to the tentative model G_t^*. Arithmetic sharing is used for these summations. Then, in 6, B sends its shares of G_t^* to A who reconstructs G_t^* and divides it by L before adding noise in plaintext. Using techniques from (Eigner et al., 2014), we can also add noise in STPC to protect the global models at the expense of higher communication and computation. Finally, the new global model G_t is sent back to the clients for the next training iteration.

D BACKDOOR ATTACKS ON FEDERATED LEARNING

The broad applicability of Federated Learning (FL), in particular in applications with a huge number of users such as next word prediction (McMahan & Ramage, 2017) or for security-critical tasks (Nguyen et al., 2019) makes it attractive for malicious behavior like backdooring (Bagdasaryan et al., 2020; Shen et al., 2016; Fung et al., 2018). In these attacks, the adversary $\mathcal{A}^c$ manipulates the local models W_i to obtain poisoned models W_i' of $K' < \frac{K}{2}$ of compromised clients which are then aggregated into the global model G_t and affect its behavior. The poisoned model G_t behaves almost normally on all inputs except for specific attacker-chosen inputs $x \in I_{\mathcal{A}^c}$ (the trigger set backdoors) for which it outputs attacker-chosen (incorrect) predictions. To backdoor FL, previous work uses *data poisoning* (Shen et al., 2016) or *model poisoning* (Bagdasaryan et al., 2020).

Data Poisoning. In this attack, $\mathcal{A}^c$ adds manipulated "poisoned" data to the training data (Shen et al., 2016; Nguyen et al., 2020) of the K' compromised clients. We denote the amount of injected poisoned data $|D_{\mathcal{A}^c}|$ with respect to the size of the overall poisoned training dataset D_i' of client i

by the *Poisoned Data Rate (PDR)*:

$$PDR = \frac{|D_{\mathcal{A}^c}|}{|D'_i|}. \tag{3}$$

$\mathcal{A}^c$ will choose a PDR that maximizes the accuracy for the injected backdoor while the malicious models $W'_1, \ldots, W'_{K'}$ remain undetected by the aggregator's anomaly detector that eliminates model updates deviating from the current global model G_{t-1} or the (benign) majority of the updates of other clients.

Model Poisoning. This more substantial threat scenario assumes that $\mathcal{A}^c$ fully controls the compromised clients and can also manipulate the training mechanism, its parameters, and scale the resulting update to maximize attack impact while evading the aggregator's deployed defenses. Bagdasaryan et al. (2020) introduced such an attack called constrain-and-scale that can circumvent state-of-the-art defenses (Fung et al., 2018; Blanchard et al., 2017; McMahan et al., 2018).

Constrain-and-scale. In a first step, $\mathcal{A}^c$ trains each of the local models W'_i with poisoned data and modifies the loss function to keep the resulting model close to the original global model G_{t-1} while still achieving a high Backdoor Accuracy. For this purpose, $\mathcal{A}^c$ combines the original loss function L_{train} (indicating the normal performance of the model on the training data) with a second loss function $L_{anomaly}$ that measures the similarity between the model W' and the benign global model G_{t-1}. The actual loss function is therefore given by:

$$L = \alpha L_{train} + (1 - \alpha) L_{anomaly}. \tag{4}$$

The parameter α weights the importance of the attack impact in comparison to the attack stealthiness. The higher α is, the more the model learns on the backdoor task, but the more the model can deviate from G_{t-1} making detection easier. In the second step, W'_i is scaled to maximize the attack impact while ensuring the Euclidean distance (cf. Def. 1 in §B) of the poisoned model remains below a specified detection threshold S in order to evade the anomaly detector of the aggregator:

$$W'_i = (W'_i - G_{t-1}) \frac{S}{\|W'_i - G_{t-1}\|} + G_{t-1}. \tag{5}$$

Previously proposed FL backdoor defenses (Fung et al., 2018; Blanchard et al., 2017; McMahan et al., 2018; Muñoz-González et al., 2019; Shen et al., 2016) can either not protect against adaptive attacks in which the adversary dynamically modifies his attack based on the applied defense, or against the simultaneous injection of more than one backdoor. We discuss these defenses, their limitations, and differences to BAFFLE in §6.

Distributed Backdoor Attack (DBA; Xie et al. (2020)). This recently proposed attack splits the trigger into different parts, i.e., uses multiple colored patches as trigger. However, compared to a centralized attack, where a backdoor is the same among malicious client, the DBA assigns each client one of these trigger parts. Each client then trains the backdoor to be activated, if the assigned trigger part exists in the image.

E Details of our Experimental Setup

E.1 Datasets and Learning Configurations

Following recent research on FL and poisoning attacks on FL, we evaluate our system in three typical application scenarios: word prediction (McMahan & Ramage, 2017; McMahan et al., 2017; 2018; Lin et al., 2018), image classification (Sheller et al., 2018a;b; Chilimbi et al., 2014), and IoT (Nguyen et al., 2019; 2020; Schneible & Lu, 2017; Ren et al., 2019; Samarakoon et al., 2018; Wang et al., 2019; Smith et al., 2017). Tab. 5 summarizes the used datasets and learning models.

Word Prediction. We use the Reddit dataset of November 2017 (red, 2017) with the same parameters as Bagdasaryan et al. (2020) and McMahan et al. (2017; 2018) for comparability. Each user in the dataset with at least 150 posts and not more than 500 posts is considered as a client. We generated a dictionary based on the most frequent 50 000 words. The model consists of two LSTM layers and a linear output layer (Bagdasaryan et al., 2020; McMahan et al., 2017). It is trained for 5,000 iterations with 100 randomly selected clients in each iteration; each client trains for 250

Table 5: Datasets used in our evaluations for word prediction (WP), image classification (IC), and network intrusion detection system (NIDS) scenarios.

Application	WP	NIDS	IC		
Datasets	Reddit	IoT-Traffic	CIFAR-10	MNIST	Tiny-ImageNet
#Records	20.6M	65.6M	60K	70K	120K
Model	LSTM	GRU	ResNet-18 Light	CNN	ResNet-18
#params	∼20M	∼507K	∼2.7M	∼431k	∼11M

epochs per iteration. The adversary uses 10 malicious clients to train backdoored models. To be comparable to the attack setting in Bagdasaryan et al. (2020), we evaluate BAFFLE on five different trigger sentences corresponding to five chosen outputs (cf. §F.6 for the results).

Image Classification. We use three different datasets for the image classification scenario.

CIFAR-10. This dataset (Krizhevsky & Hinton, 2009) is a standard benchmark dataset for image classification, in particular for FL (McMahan et al., 2017) and backdoor attacks (Bagdasaryan et al., 2020; Baruch et al., 2019; Muñoz-González et al., 2019). It consists of 60 000 images of 10 different classes. The adversary aims at changing the predicted label of one class of images to another class of images. Bagdasaryan et al. (2020) experiment with a backdoor where *green cars* are predicted to be *birds*, but we extend our evaluation to different backdoors, e.g., cats that are incorrectly labeled as airplanes (cf. §F.7). We use a lightweight version of the ResNet18 model (He et al., 2016) with 4 convolutional layers with max-pooling and batch normalization (Bagdasaryan et al., 2020).

MNIST. The MNIST dataset consists of 70 000 handwritten digits (LeCun et al., 1998). The learning task is to classify images to identify digits. The adversary poisons the model by mislabeling labels of digit images before using it for training (Shen et al., 2016). We use a convolutional neural network (CNN) with

Tiny-ImageNet. Tiny-ImageNet[3] consists of 200 classes and each class has 500 training images, 50 validation images, and 50 test images. For Tiny-ImageNet, we used ResNet18 He et al. (2016) as model.

Network Intrusion Detection System (NIDS). We test backdoor attacks on IoT anomaly-based intrusion detection systems that often represent critical security applications (Antonakakis et al., 2017; Herwig et al., 2019; Doshi et al., 2018; Soltan et al., 2018; Kolias et al., 2017; Nguyen et al., 2019; 2020). Here, the adversary aims at causing incorrect classification of anomalous traffic patterns, e.g., generated by IoT malware, as benign patterns. Based on the FL anomaly detection system DÏoT by Nguyen et al. (2019), we use three datasets shared by Nguyen et al. (2019) and Sivanathan et al. (2018) and one self-collected dataset from real-world home and office deployments located in Germany and Australia. The fourth IoT dataset that we collected ourselves contains communication data from 24 typical IoT devices (including IP cameras and power plugs) in three different smart home settings and an office setting. Tab. 6 provides the details of all four IoT datasets used in our experiments. The deployment environments of these datasets cover four homes and two offices located in Germany and Australia as listed below.

Table 6: Characteristics of IoT datasets

Dataset	No. devices	Time (hours)	Size (MB)	Packets (millions)
BAFFLE-*Benign*	28	7 603	1 153	6.4
DIoT-Benign	18	2 352	578	2.3
UNSW-Benign	27	7 457	11 759	23.9
DIoT-Attack	5	80	7 734	21.9

1. BAFFLE-*Benign*: Traffic that we captured from 28 IoT devices in three smart home settings and an office setting. The smart home settings consist of two flats and one house in different cities, with 1 to 4 inhabitants. The office setting is a 20 m^2 office for two people. In each experiment, we deployed 28 IoT devices for more than one week and encouraged users to use the devices as part of their daily activities.

[3]https://tiny-imagenet.herokuapp.com

2. *DIoT-Benign*: Traffic that was captured from 18 IoT devices deployed in a real-word smart home (Nguyen et al., 2019).
3. *UNSW-Benign*: Traffic that was captured from 28 IoT devices in an office for 20 days (Sivanathan et al., 2018).
4. *DIoT-Attack*: Traffic generated by 5 IoT devices infected by the Mirai malware (Nguyen et al., 2019).

Following Nguyen et al. (2019), we extracted device-type-specific datasets capturing the devices' communication behavior. Thereby, we prioritize device types that are present in several datasets and have sufficient data for evaluating them in a simulated FL setting where the data has to be split among the clients, i.e., *Security Gateway*s. In total, we evaluate BAFFLE on data from 50 devices of 24 device types. We simulate the FL setup by splitting each device type's dataset among several clients (from 20 to 200). Each client has a training dataset corresponding to three hours of traffic measurements containing samples of roughly 2 000-3 000 communication packets. We extensively evaluate BAFFLE on all 13 backdoors corresponding to 13 Mirai's attacks (cf. §F.5 for details). However, by IoT-Traffic dataset we denote a subset that contains data collected with the NetatmoWeather device type (a smart weather station). The model consists of 2 GRU layers and a fully connected output layer.

E.2 EVALUATION METRICS

We consider a set of metrics for evaluating the effectiveness of backdoor attack and defense techniques:

- **BA - Backdoor Accuracy** indicates the accuracy of the model in the backdoor task, i.e., it is the fraction of the trigger set for which the model provides the wrong outputs as chosen by the adversary. The adversary aims to maximize BA.
- **MA - Main Task Accuracy** indicates the accuracy of a model in its main (benign) task. It denotes the fraction of benign inputs for which the system provides correct predictions. The adversary aims at minimizing the effect on MA to reduce the chance of being detected. The defense system should not negatively impact MA.
- **PDR - Poisoned Data Rate** refers to the fraction of poisoned data in the training dataset. Using a high PDR can increase the BA but is also likely to make poisoned models more distinguishable from benign models and thus easier to detect.
- **PMR - Poisoned Model Rate** is the fraction of poisoned models.
- **TPR - True Positive Rate** indicates how well the defense identifies poisoned models, i.e., the ratio of the number of models correctly classified as poisoned to the total number of models classified as poisoned.
- **TNR - True Negative Rate** indicates the ratio of the number of local models correctly classified as benign to the total number of models classified as benign. The higher the TNR, the less poisoned models are aggregated in the global model.

F EXTENDED EXPERIMENTAL EVALUATION

F.1 EFFECTIVENESS OF EACH OF BAFFLE'S COMPONENTS

In this section, we separately evaluate the effectiveness of each of BAFFLE's components.

Effectiveness of the Clustering. We show the results for the clustering in Tab. 7. As shown there, our clustering achieves $TNR = 100\%$ for the Reddit and IoT-Traffic datasets, i.e., BAFFLE only selects benign models in this attack setting. For the CIFAR-10 dataset, TNR is not maximal (86.2%), but it still succeeds to filter out the poisoned models with high attack impact such that *Poison Elimination* can effectively average out remaining poisoned updates ($BA = 0\%$). Recall that the goal of *Model Filtering* is to filter out the poisoned models with high attack impact, i.e., not necessarily all poisoned models (cf. §3).

Table 7: Effectiveness of the clustering component, in terms of True Positive Rate (TPR) and True Negative Rate (TNR), of BAFFLE in comparison to existing defenses for the constrain-and-scale attack on three datasets. All values are in percentage and the best results of the defenses are marked in bold (cf. §E.2 for detailed information about the metrics).

Defenses	Reddit		CIFAR-10		IoT-Traffic	
	TPR	TNR	TPR	TNR	TPR	TNR
Krum	9.1	0.0	8.2	0.0	24.2	0.0
FoolsGold	**100.0**	**100.0**	0.0	90.0	32.7	84.4
Auror	0.0	90.0	0.0	90.0	0.0	70.2
AFA	0.0	88.9	**100.0**	**100.0**	4.5	69.2
BAFFLE	22.2	**100.0**	23.8	86.2	**59.5**	**100.0**

Impact of the Degree of non-Independent and Identically Distributed (non-IID) Data. Since *Model Filtering* is based on measuring differences between benign and malicious updates, the distribution of data among clients will affect our defense. For CIFAR-10, we vary the degree of non-IID data, denoted by Deg_{nIID}, following previous work (Fang et al., 2020) by varying the fraction of images belonging to a specific class assigned to a specific group of clients. In particular, we divide the clients into 10 groups corresponding to the 10 classes of CIFAR-10. The clients of each group are assigned to a fixed fraction of Deg_{nIID} of the images from its designated image class, while the rest of the images will be assigned to it at random. Consequently, the data distribution is random, i.e., completely IID if $Deg_{nIID} = 0\%$ (all images are randomly assigned) and completely non-IID if $Deg_{nIID} = 100\%$ (a client only gets images from its designated class). For the Reddit and IoT datasets, changing the degree of non-IID data is not meaningful since the data has a natural distribution as every client obtains data from different Reddit users or traffic chunks from different IoT devices. To summarize, our clustering approach provides almost identical results for different values of Deg_{nIID} as TNR and TPR remain steady ($100.0\% \pm 0.00\%$ and $40.81\% \pm 0,00\%$), while BA remains at 0% and MA is $91.9\%(\pm 0.02\%)$ for all experiments.

Effectiveness of Clipping. Fig. 7 demonstrates the effectiveness of BAFFLE's dynamic clipping where S is the L_2-norm median compared to a static clipping (Bagdasaryan et al., 2020). Fig. 7a and Fig. 7b show that a small static bound $S = 0.5$ is effective to mitigate the attack ($BA = 0\%$), but MA drops to 0% rendering the model inoperative. Moreover, a higher static bound like $S = 10$ is ineffective as $BA = 100\%$ if the Poisoned Data Rate (PDR) $\geq 35\%$. In contrast, BAFFLE's dynamic clipping threshold performs significantly better (cf. Fig. 7c and Fig. 7d). Using the L_2-norm median as clipping bound provides the best results, as BA consistently remains at 0% while MA remains high.

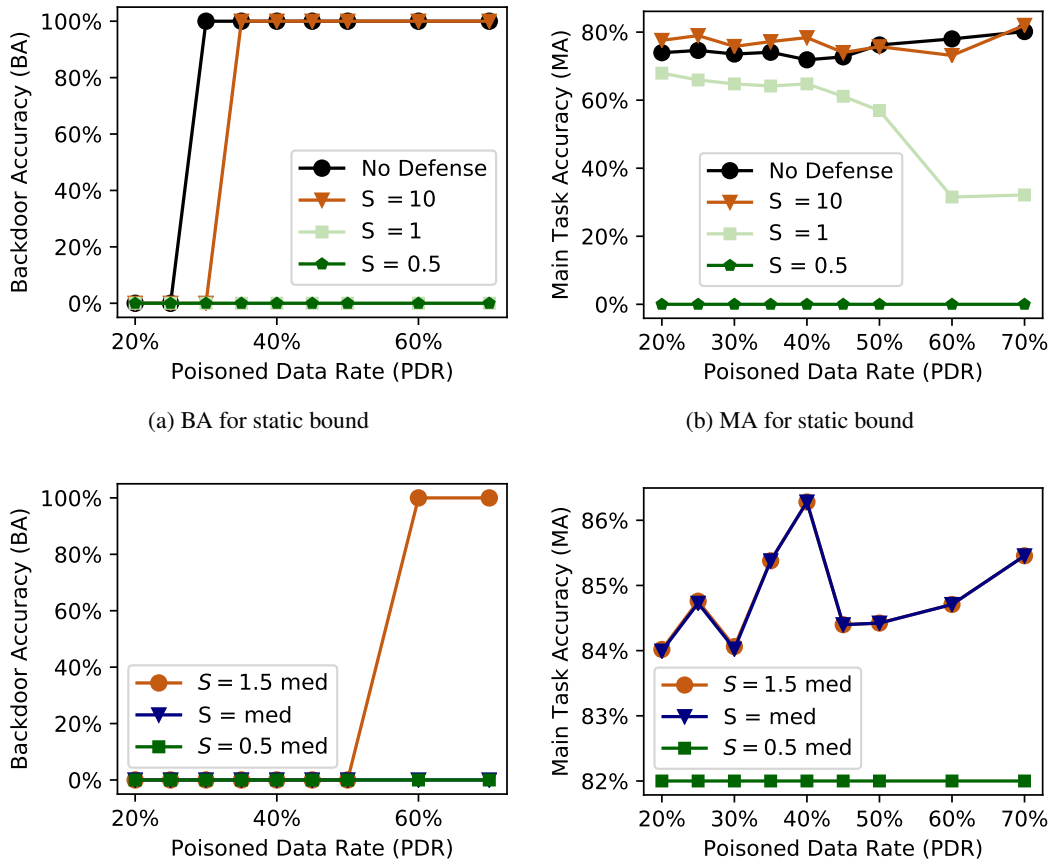
Effectiveness of Adding Noise. Fig. 8 shows the impact of adding noise to the intermediate global models with respect to different noise level factors λ. As it can be seen, increasing λ reduces the BA, but it also negatively impacts the performance of the model in the main task (MA). Therefore, the noise level must be dynamically tuned and combined with the other defense components to optimize the overall success of the defense.

Furthermore, we test a naïve combination of the defense layers by stacking clipping and adding noise (using a fixed clipping bound of 1.0 and a standard deviation of 0.01 as in Bagdasaryan et al. (2020)) on top of a filtering layer using K-means. However, this naïve approach still allows a BA of 51.9% and a MA of 60.24%, compared to a BA of 0.0% and a MA of 89.87% of BAFFLE in the same scenario. Based on our evaluations in §5.1, it becomes apparent that BAFFLE's dynamic nature goes beyond previously proposed defenses that consist of static baseline ideas, which BAFFLE significantly optimizes, extends, and automates to offer a comprehensive dynamic and private defense against sophisticated backdoor attacks.

F.2 RESILIENCE TO ADAPTIVE ATTACKS

Given sufficient knowledge about BAFFLE, an adversary may seek to use adaptive attacks to bypass the defense layers. In this section, we analyze such attack scenarios and strategies including *changing the injection strategy*, *model alignment*, and *model obfuscation*.

Changing the Injection Strategy. The adversary may attempt to simultaneously inject several backdoors in order to execute different attacks on the system in parallel or to circumvent the clus-



(a) BA for static bound

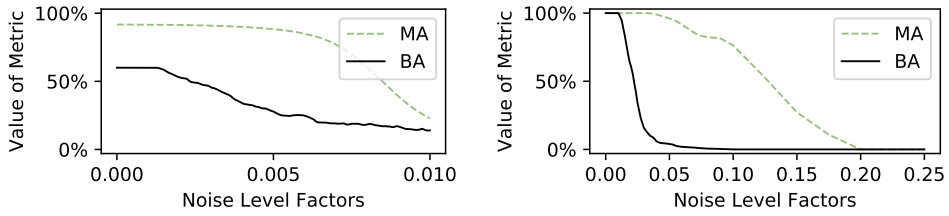
(b) MA for static bound

(c) BA for dynamic bound

(d) MA for dynamic bound

Figure 7: Effectiveness, in terms of Backdoor Accuracy (BA) and Main Task Accuracy (MA), of BAFFLE's dynamic clipping bound. S is the clipping bound and med the L_2-norm median.

tering defense (cf. §2). BAFFLE is also effective against such attacks (cf. Fig. 2 on p. 4). To further investigate the resilience of BAFFLE against such attacks, we conduct two experiments: (1) assigning different backdoors to malicious clients and (2) letting a malicious client inject several backdoors. We conduct these experiments with $K = 100$ clients of which $K' = 40$ are malicious on the IoT-Traffic dataset with each type of Mirai attack representing a backdoor. In the first experiment, we evaluate BAFFLE for $0, 1, 2, 4$, and 8 backdoors meaning that the number of malicious clients for each backdoor is $0, 40, 20, 10$, and 5. Our experimental results show that our approach is



(a) Image Classification

(b) Network Intrusion Detection

Figure 8: Impact of different noise level factors on the Backdoor Accuracy (BA) and Main Task Accuracy (MA) (cf. §E.2 for detailed information about the metrics).

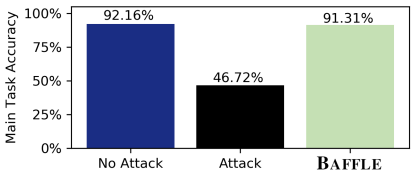


Figure 9: Resilience of BAFFLE against an untargeted poisoning attack in terms of Main Task Accuracy (MA).

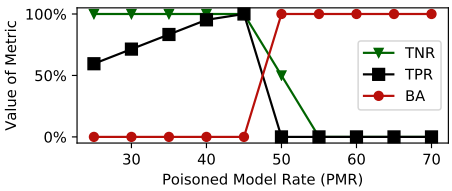


Figure 10: Impact on the evaluation metrics of the poisoned model rate $PMR = \frac{K'}{K}$ which is the fraction of malicious clients K' per total clients K (cf. §E.2 for detailed information about the metrics).

effective in mitigating the attacks as $BA = 0\% \pm 0.0\%$ in all cases, with $TPR = 95.2\% \pm 0.0\%$, and $TNR = 100.0\% \pm 0.0\%$. For the second experiment, 4 backdoors are injected by each of the 40 malicious clients. Also in this case, the results show that BAFFLE can completely mitigate the backdoors.

Model Alignment. Using the same attack parameter values, i.e., PDR or α (cf. §D), for all malicious clients can result in a gap between poisoned and benign models that can be separated by *Model Filtering*. Therefore, a sophisticated adversary can generate models that bridge the gap between them such that they are merged to the same cluster in our clustering. We evaluate this attack on the IoT-Traffic dataset for $K' = 80$ malicious clients and $K = 200$ clients in total. To remove the gap, each malicious client is assigned a random amount of malicious data, i.e., a random PDR ranging from 5% to 20%. Tab. 8 shows the effectiveness of BAFFLE against such attacks. Although BAFFLE cannot cluster the malicious clients well ($TPR = 5.68\%$), it still mitigates the attack successfully (BA reduces from 100% to 0%). This can be explained by the fact that when the adversary tunes malicious updates to be close to the benign ones, the attack's impact is reduced and consequently averaged out by *Poison Elimination*.

Model Obfuscation. The adversary can add noise to the poisoned models to make them difficult to detect. However, our evaluation of such an attack on the IoT-Traffic dataset shows that this strategy is not effective. We evaluate different noise levels to determine a suitable standard deviation for the noise. Thereby, we observe that a noise level of 0.034 causes the models' Cosine distances in clustering to change without significantly impacting BA. However, BAFFLE can still efficiently defend this attack: BA remains at 0% and MA at 100%.

F.3 Impact of Number of Malicious Clients

We assume that more than half of all clients are benign (cf. §2) and our clustering is only expected to be successful when $PMR = \frac{K'}{K} < 50\%$ (cf. §3.1). We evaluate BAFFLE for different PMR values. Fig. 10 shows how BA, TPR, and TNR change in the NIDS application depending on PMR values from 25% to 75%. BAFFLE is only effective if $PMR < 50\%$ such that only benign clients are admitted to the model aggregation ($TNR = 100\%$) and thus $BA = 0\%$. However, if $PMR > 50\%$, BAFFLE fails to mitigate the attack because all malicious models will be included ($TPR = 0\%$).

Table 8: Resilience to model alignment attacks in terms of Backdoor Accuracy (BA), Main Task Accuracy (MA), True Positive Rate (TPR), True Negative Rate (TNR) in percent (for detailed information about the metrics cf. §E.2).

	BA	MA	TPR	TNR
HDBSCAN	100.0	91.98	0.0	33.04
BAFFLE	**0.0**	**100.0**	**5.68**	**33.33**

F.4 Resilience to Untargeted Poisoning

Another attack type related to backdooring is *untargeted poisoning* resembling a denial of service (DoS) (Fang et al., 2020; Blanchard et al., 2017; Baruch et al., 2019). Unlike backdoor attacks

that aim to incorporate specific backdoor functionalities, untargeted poisoning aims at rendering the model unusable. The adversary uses crafted local models with low Main Task Accuracy to damage the global model G. Fang et al. (2020) propose such an attack bypassing state-of-the-art defenses. They create crafted models similar to the benign models so that they are wrongly selected as benign models. Although we do not focus on untargeted poisoning, our approach intuitively defends it since, in principle, this attack also trade-offs attack impact against stealthiness.

To evaluate the effectiveness of BAFFLE against untargeted poisoning, we test the sophisticated attack proposed by Fang et al. (2020) on BAFFLE. The authors introduce three attacks against different aggregation rules: Krum (Blanchard et al., 2017), Trimmed Mean, and Median (Yin et al., 2018). Among those three attacks, we consider the Krum-based attack because it: (1) is the focus of their work and stronger than the others, (2) can be transferred to unknown aggregation rules, and (3) has a formal convergence proof (Blanchard et al., 2017; Fang et al., 2020). Since Fang et al. (2020)'s evaluation uses image datasets, we evaluate BAFFLE's resilience against it with CIFAR-10. Fig. 9 demonstrates BAFFLE's effectiveness against these untargeted poisoning attacks. It shows that although the attack significantly damages the model by reducing MA from 92.16% to 46.72%, BAFFLE can successfully defend against it and MA remains at 91.31%.

F.5 EFFECTIVENESS OF BAFFLE FOR DIFFERENT MIRAI ATTACK TYPES

To evaluate the performance of BAFFLE against different backdoors (in this case, different *Mirai* attacks), we take all 13 attack types available in the attack dataset (Nguyen et al., 2019) and try to inject them as backdoors. The adversary controls 25 out of 100 clients and uses a PDR of 50%. For each backdoor, the adversary applies the Constrain-and-scale attack (cf. §D) for 5 rounds, while BAFFLE is used as defense. Tab. 9 shows the results. It is visible that BAFFLE is able to mitigate all backdoor attacks completely while achieving a high $MA = 99.8\%$.

Table 9: Comparison of the Backdoor Accuracy (BA) when injecting different backdoors while using (1) Poison Elimination, (2) Clustering, and (3) BAFFLE as defense (Main Task Accuracy $MA = 99.8\%$ for all cases in BAFFLE).

Backdoor	Baseline	(1)	(2)	(3)
Dos-Ack	100.0%	53.5%	0.0%	0.0%
Dos-Dns	100.0%	17.9%	0.0%	0.0%
Dos-Greeth	100.0%	19.3%	0.0%	0.0%
Dos-Greip	100.0%	59.8%	0.0%	0.0%
Dos-Http	100.0%	24.1%	0.0%	0.0%
Dos-Stomp	100.0%	95.0%	100.0%	0.0%
Dos-Syn	100.0%	13.5%	0.0%	0.0%
Dos-Udp	100.0%	40.0%	0.0%	0.0%
Dos-Udp (Plain)	100.0%	100.0%	0.0%	0.0%
Dos-Vse	100.0%	54.9%	0.0%	0.0%
Infection	17.0%	4.3%	25.4%	0.0%
Preinfection	50.2%	7.4%	0.0%	0.0%
Scan	100.0%	46.9%	0.0%	0.0%
Average	89.8%	41.3%	9.6%	0.0%

F.5.1 EFFECTIVENESS OF BAFFLE FOR DIFFERENT DEVICE TYPES

Tab. 10 shows the effectiveness of BAFFLE and each of its individual components compared to the baseline where no defense measures are used. Analogous to the experiments in Tab. 9, the adversary controls 25% of the clients and uses a PDR of 50% for running the Constrain-and-scale attack (cf. §D) to inject a backdoor for the Mirai scanning attack. The attack is run for 3 training iterations. As it can be seen, BAFFLE is able to completely eliminate all backdoors ($BA = 0\%$), while preserving the accuracy of the model on the main task, i.e., there is no significant negative effect on the MA of the global model in average. Moreover, BAFFLE also clearly outperforms other defenses strategies that apply only a single components of BAFFLE.

Table 10: Backdoor Accuracy (BA) and Main Task Accuracy (MA) when applying (1) Poison Elimination, (2) Clustering, and (3) BAFFLE as defense.

Device Type	BA				MA			
	Baseline	(1)	(2)	(3)	Baseline	(1)	(2)	(3)
AmazonEcho	100.0%	43.3%	0.0%	0.0%	99.5%	91.6%	100.0%	97.1%
DLinkCam	100.0%	47.6%	0.0%	0.0%	99.7%	98.5%	97.5%	89.9%
DLinkType05	100.0%	44.2%	0.0%	0.0%	84.7%	76.9%	98.7%	94.2%
EdimaxPlug	100.0%	24.1%	0.0%	0.0%	99.3%	98.0%	99.3%	97.6%
EdnetGateway	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
GoogleHome	100.0%	87.1%	0.0%	0.0%	100.0%	94.7%	100.0%	99.9%
HPPrinter	100.0%	100.0%	0.0%	0.0%	86.6%	85.2%	68.0%	68.0%
iHome	100.0%	100.0%	0.0%	0.0%	93.1%	93.1%	93.3%	93.2%
LiFXSmartBulb	100.0%	92.2%	0.0%	0.0%	94.3%	96.6%	93.5%	93.4%
Lightify2	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
NestDropcam	100.0%	62.4%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
NetatmoCam	100.0%	60.0%	100.0%	0.0%	99.2%	98.7%	99.3%	97.5%
NetatmoWeather	100.0%	94.6%	100.0%	0.0%	100.0%	100.0%	99.6%	100.0%
PIX-STARPhoto	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
RingCam	100.0%	86.8%	0.0%	0.0%	96.1%	95.0%	96.1%	95.4%
SamsungSmartCam	100.0%	85.3%	0.0%	0.0%	100.0%	99.5%	100.0%	99.7%
Smarter	100.0%	100.0%	0.0%	0.0%	93.3%	93.3%	100.0%	100.0%
SmartThings	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
TesvorVacuum	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
TP-LinkCam	100.0%	100.0%	0.0%	0.0%	67.2%	67.2%	67.1%	67.0%
TPLinkPlug	100.0%	100.0%	0.0%	0.0%	97.7%	96.5%	99.9%	98.6%
TribySpeaker	100.0%	100.0%	0.0%	0.0%	95.3%	90.7%	88.7%	76.9%
WithingsSleepS	100.0%	100.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%
WithingsBabyM	100.0%	80.0%	100.0%	0.0%	100.0%	100.0%	56.2%	100.0%
Average	100.0%	83.7%	12.5%	0.0%	96.1%	94.8%	94.0%	94.5%

F.6 PERFORMANCE OF BAFFLE FOR DIFFERENT NLP BACKDOORS

To demonstrate BAFFLE's general applicability, we use it to defend backdoor attacks on a next word prediction task with multiple different backdoors as shown in Tab. 11:
(1): "delicious" after the sentence "pasta from astoria tastes"
(2): "bing" after the sentence "search online using"
(3): "expensive" after the sentence "barbershop on the corner is"
(4): "nokia" after the sentence "adore my old"
(5): "rule" after the sentence "my headphones from bose"

Table 11: Main Task Accuracy (MA), Backdoor Accuracy (BA), True Positive Rate (TPR), and True Negative Rate (TNR) of BAFFLE for different NLP backdoors (all values in percentage).

	No Defense		BAFFLE			
Backdoor	BA	MA	BA	MA	TPR	TNR
"delicious"	100.0	22.6	0.0	22.3	22.2	100.0
"bing"	100.0	22.4	0.0	22.3	20.4	100.0
"expensive"	100.0	22.2	0.0	22.3	20.4	100.0
"nokia"	100.0	22.4	0.0	22.0	20.4	100.0
"rule"	100.0	22.3	0.0	22.0	20.4	100.0
Average	100.0	22.4	0.0	22.2	20.8	100.0

F.7 PERFORMANCE OF BAFFLE FOR DIFFERENT IMAGE BACKDOORS

To demonstrate BAFFLE's general applicability and evaluate its performance in wider attack scenarios than the very specific backdoor of Bagdasaryan et al. (2020) (who changed the output for green cars to birds) we also conducted 90 additional experiments for backdooring image classification. In these experiments, we test on all possible pairs of instances and try to change the predictions of one class to each other possible class. Here, BAFFLE reduces the attack impact from $BA = 53.92 \pm 27.51$ to $BA = 2.52 \pm 5.83$ in average. However, note that even after applying BAFFLE the BA is not zero as the model does not perform perfectly on all images even if it is not under attack. Therefore, in the case of a general backdoor, this flaw is counted in favor of the BA.

F.8 EVALUATION OF BAFFLE AGAINST DBA

Table 12: Parameter setup for the evaluation of BAFFLE against the DBA.

	CIFAR-10	MNIST	Tiny-ImageNet
Number of Pretrained Rounds	200	10	20
Rounds without Attack	2	1	0
Local Epochs of Benign Clients	2	1	2
Local Epochs of Malicious Clients	6	10	10
Learning Rate of Benign Clients	10^{-1}	10^{-1}	10^{-3}
Learning Rate of Malicious Clients	$5 * 10^{-2}$	$5 * 10^{-2}$	10^{-3}

We evaluated BAFFLE in the same setup as used by Xie et al. (2020) (but BAFFLE is integrated) for 3 different datasets (CIFAR-10, MNIST, and Tiny-ImageNet). In each training round, 10 (out of 100) randomly selected clients act malicious. Following the setup of Xie *et al.*, we used a model that was trained only on benign clients and continued the training for some rounds in case of the CIFAR-10 and MNIST dataset with our BAFFLE being deployed, before launching the attack. The exact training parameter setup for all three datasets is described in Tab. 12.

Table 13: Main Task Accuracy (MA) and Backdoor Accuracy (BA) of BAFFLE against the DBA (all values in percentage).

	CIFAR-10				MNIST				Tiny-ImageNet			
	No Defense		BAFFLE		No Defense		BAFFLE		No Defense		BAFFLE	
	BA	MA	BA	MA	BA	MA	BA	MA	BA	MA	BA	MA
Pretrained Model	2.2	75.9	2.2	75.9	0.5	97.2	0.5	97.2	0.1	56.5	0.1	56.5
Before First Attack	2.4	77.4	2.4	76.0	0.5	97.3	0.5	97.2	0.1	56.5	0.1	56.5
After Attack	93.8	57.4	3.2	76.2	99.3	87.9	0.5	97.3	97.0	16.3	0.1	56.4

Tab. 13 contains the results of the DBA when deploying BAFFLE compared to the baseline scenario where no defense is deployed. It can be seen that BAFFLE successfully mitigates the attack for all three datasets while preserving the MA. However, the BA is not 0% even before the attack because the model mislabels some images (as the MA is not 100%) and this mislabeling is counted in favor for the BA when the predicted label is equal to the target label by chance.

F.9 COMMUNICATION OF PRIVATE BAFFLE

While in traditional FL each client sends its model to the server and later receives the aggregated model, in private BAFFLE (cf. §3 and §C), each client has to sent shares of its model to the two servers, and receives one aggregated model at the end. In addition, the communication in private BAFFLE is done using 64-bit fixed point numbers, while PyTorch uses 32-bit floating point numbers. Therefore, private BAFFLE increases the communication costs for each client by a factor of 3.

In addition, also both aggregation servers need to communicate with each other. Tab. 14 shows the communication costs of the servers in GB caused by using STPC for Cosine distance calculation, clustering, and Euclidean distance calculation/clipping/aggregating in each update iteration of FL. As the computation is done between two servers, we can assume a well-connected network with high throughput and low latency such that this overhead is acceptable.

Table 14: Communication in GB of private BAFFLE's Cosine distances and of the Euclidean distances/clipping/model aggregation with different numbers of accepted models $\frac{K}{2} + 1$ and different applications/model sizes. K is the number of clients and clustering is independent of the model size.

K	Cosine Distance			Euclidean Distance + Clipping + Model Aggregation			Clustering
	Reddit	CIFAR-10	IoT-Traffic	Reddit	CIFAR-10	IoT-Traffic	
10	202	110	91	128	54	45	0.2
50	2 527	248	125	220	70	60	7.0
100	9 598	586	235	601	132	68	38