Poisoning Attacks on Federated Learning for Autonomous Driving

Sonakshi Garg[1,2], Hugo Jönsson[2,3], Gustav Kalander[2,4], Axel Nilsson[2,3], Bhhaanu Pirange[2,5], Viktor Valadi[2,6], and Johan Östman[2]

***Abstract*— Federated Learning (FL) is a decentralized learning paradigm, enabling parties to collaboratively train models while keeping their data confidential. Within autonomous driving, it brings the potential of reducing data storage costs, reducing bandwidth requirements, and to accelerate the learning. FL is, however, susceptible to poisoning attacks. In this paper, we introduce two novel poisoning attacks on FL tailored to regression tasks within autonomous driving: FLStealth and Off-Track Attack (OTA). FLStealth, an untargeted attack, aims at providing model updates that deteriorate the global model performance while appearing benign. OTA, on the other hand, is a targeted attack with the objective to change the global model's behavior when exposed to a certain trigger. We demonstrate the effectiveness of our attacks by conducting comprehensive experiments pertaining to the task of vehicle trajectory prediction. In particular, we show that, among five different untargeted attacks, FLStealth is the most successful at bypassing the considered defenses employed by the server. For OTA, we demonstrate the inability of common defense strategies to mitigate the attack, highlighting the critical need for new defensive mechanisms against targeted attacks within FL for autonomous driving.**

I. Introduction

Machine learning models deployed in-car are typically trained centrally on vast amounts of collected data [1]. However, centrally stored data is subject to large costs and may be subject to privacy concerns in relation to, e.g., the GDPR [2]. Further, in the case of wireless data collection, the data transmission requires significant bandwidth. To remedy these shortcomings, federated learning (FL) has been proposed as a potential solution. The main idea of FL is to train machine learning models locally, thereby maintaining data confidentiality, and then aggregate the locally trained models centrally into a global model [3]. Several FL frameworks, tailored for autonomous driving, have recently been introduced [4], [5], [6].

Within the automotive sector, companies like Toyota and Ford are exploring FL solutions across various applications, e.g., object detection [7] and turn-signal prediction [8]. As vehicular networks are intrinsically dynamic, a recent direction of research also pertains to developing novel protocols for the selection of vehicle within the federation [9]. However, as control is moved from a central entity to the vehicles, new attack surfaces emerge. For example, a given vehicle may manipulate their local model towards a malicious objective, referred to as a poisoning attack, which could ultimately result in traffic accidents. Hence, in any FL application, it is imperative to provide defences against vehicles with devious intentions. A common mitigation strategy to such attacks is to employ robust aggregation of local models where the impact of outliers is limited [10], [11].

From the adversary perspective, poisoning attacks on FL are commonly tailored towards classification problems [12], [13] with only a small number targeting regression problems [14], [15]. However, regression tasks are common in autonomous driving, e.g., vehicle speed prediction, distance estimation, time-to-collision prediction, and vehicle trajectory prediction. Therefore, in this paper, we investigate poisoning attacks on FL for regression tasks within autonomous driving. We introduce two attacks coined FLSTEALTH and Off-Track Attack (OTA). The former is a general untargeted attack with the objective to deteriorate the global model performance whereas the latter is a backdoor attack tailored specifically to the problem of vehicle trajectory prediction. We conduct an experimental study, using the Zenseact Open Dataset (ZOD) [16], on the impact of untargeted attacks on vehicle trajectory prediction and to what extent common defenses are effective. Furthermore, by using OTA, we demonstrate that FL systems are vulnerable to targeted attacks and that they may significantly impact the behavior of the global model. Notably, common defense mechanism are largely inefficient against OTA.

II. Preliminaries

A. Federated Learning

Federated learning (FL) is a learning paradigm where multiple clients collaboratively train a model without revealing their local data [3]. In particular, FL attempts to find a model $\theta^\star$ according to

$$\theta^\star = \arg\min_\theta \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{(x,y)\sim\mathcal{P}_i}\left[\ell(x,y;\theta)\right] \tag{1}$$

where n is the number of clients in the federation, $\ell(x,y;\theta)$ denotes the loss function, parameterized by the model θ, evaluated on a sample (x,y), $\mathcal{P}_i$ denotes the local data distribution of client $i \in [n]$, and $\mathbb{E}[\cdot]$ is used for expectation. Practically, the expectation is approximated locally by the sample average over a training dataset D_i sampled from $\mathcal{P}_i$.

To solve (1), a server coordinates multiple clients over several rounds, each initiated by broadcasting a global model. The server then collects a locally updated version of the

[1] Umeå University
[2] AI Sweden
[3] Royal Institute of Technology
[4] Chalmers University of Technology
[5] Dakota State University
[6] Scaleout Systems

broadcasted model from the clients and aggregates it into an updated global model. This iterative procedure proceeds until the global model converges or a predefined number of training rounds is reached.

B. Poisoning Attacks in Federated Learning

FL is vulnerable to clients with malicious intent that may manipulate their local updates before sending it to the server, so-called poisoning attacks. Such attacks are multifaceted and may be untargeted [17], [18], i.e., aim to deteriorate the global model performance, or targeted, i.e., alter the behavior of the global model on specific data samples [12], [19], [20]. Poisoning attacks may be divided into data poisoning [21], [22], [23] and model poisoning [18], [24], [25], [26] where the former alters the underlying dataset and the latter directly manipulates the model weights. It should be noted that any data poisoning attack can be replicated using a model poisoning attack.

Some common untargeted attacks include label flipping, gradient ascent attacks, and model shuffling. In a label flipping attack, the attacker intentionally alters the labels within its dataset to prevent the global model from learning patterns in the data [27], [22]. In gradient ascent attacks, the attacker updates the model in the direction that maximizes the loss. The model shuffling attack aims at shuffling the model parameters without notably changing the loss [28].

Backdoor attacks typically rely on triggers injected in the data, causing the model to misbehave when exposed to the trigger [23], [20]. An example pertaining to street-sign detection is given in [29]. Therein, a street-sign detector typically performs well but may incorrectly identify stop signs with a particular sticker as speed limit signs. Such a behavior can be achieved by the following optimization procedure

$$\theta^\star = \arg\min_\theta \sum_{(x,y)\in D_\mathrm{H}} \ell(x,y;\theta) + \sum_{(x,y)\in D_\mathrm{B}} \ell(\mu(x,y);\theta) \tag{2}$$

where D_H denotes an honest dataset and D_B a byzantine dataset to be used for the backdoor attack. Samples in D_B are manipulated using some perturbation mechanism μ aligned with the backdoor objective. Notably, a backdoor attack aligns with the global objective on the honest dataset.

C. Poisoning Mitigation Strategies in Federated Learning

Any convincing defensive mechanism should be able to handle an arbitrary attack. For this reason, the byzantine threat model, allowing an attacker to directly alter the model weights to submit arbitrary updates, is prevalent. Within byzantine resilient FL, there are two categories: robust aggregation [10], [30], [31] and anomaly detection [32]. The former category is based on outlier mitigation, i.e., it relies on benign clients submitting similar models, whereas the latter category attempts to directly identify misbehaving clients. In this paper, we shall focus on the former class of strategies.

A non-exhaustive list of robust aggregation techniques include KRUM [10], FLTRUST [11], TRIMMEDMEAN [30], PCA Defence [22], loss-function based rejection (LFR) [25] and Loss Defence. The first four methods relies on benign clients being similar to each other or to a server-based model whereas the last two removes clients that have a large impact on the global loss obtained via a server-based validation dataset.

III. NOVEL ATTACKS ON REGRESSION TASKS

In this section, our threat model is defined and two novel attacks, pertaining to regression tasks in autonomous driving, are introduced.

A. Threat Model

We consider a federation with an honest-but-curious server and n clients out of which $m < n$ are compromised.[1] The m malicious clients may collude to perform coordinated attacks. Furthermore, the malicious clients may perform either data or model-poisoning attacks.

B. FLSTEALTH

Based on the threat model, we now introduce a novel untargeted attack on federated regression tasks. To circumvent any defensive efforts, the attack attempts to deteriorate the global model as much as possible while remaining stealthy. This is achieved by creating two models, an honest and a byzantine, both initialized from the global model. The attack is divided in two steps where the first accounts to training the honest model according as if the client was benign. Thereafter, the byzantine model is trained to maximize the loss while remaining close to the honest model. The resulting loss function of the byzantine model is given as

$$\ell_{\mathrm{FLStealth}}(x,y,\theta_\mathrm{H};\theta_\mathrm{B}) = -\kappa\ell(x,y,\theta_\mathrm{B}) + \mathrm{MSE}(\theta_\mathrm{H},\theta_\mathrm{B}) \tag{3}$$

where $\kappa \geq 0$ is a weighting constant, θ_i, $i \in \{\mathrm{H},\mathrm{B}\}$, denotes the honest and byzantine models, and MSE is the mean-squared error. As can be seen, a lower κ results in a byzantine model closer to the honest model.

C. Off-Track Attack

Next, we propose a novel backdoor attack crafted for vehicle trajectory prediction. It is based on the principle of triggers, as discussed in [23], [20], but adapted towards the specific use-case of vehicle trajectory prediction. For classification tasks, a backdoor attack can be as simple as flipping a class label. However, for regression tasks there are no classes, hence, the target has to be altered differently. In trajectory prediction, the target trajectory may be altered by slightly changing points resulting in an alternative path. The details are presented in Section V.

IV. FEDERATED VEHICLE TRAJECTORY PREDICTION

A. Dataset

We utilize the Zenseact Open Dataset (ZOD) [16], a multi-modal autonomous driving dataset collected over a period of 2 years across 14 European countries. The dataset contains

[1]We will refer to vehicles and clients interchangeably in the remainder of the paper.

three subsets: *frames* that are primarily suitable for non-temporal perception tasks, *sequences* that are intended for spatio-temporal learning and prediction, and *drives* that are aimed at longer-term tasks such as localization, mapping, and planning. The *frames* consists of more than 100k traffic scenes that have been carefully curated to cover a wide range of real-world driving scenarios. From the original 100K images in the ZOD-dataset, only 80k images were usable after filtering for missing, incomplete, or erroneous data. For each frame, the dataset contains annotations, calibration data, blurred and Deep Natural Anonymization Technology (dnat) images, ego-motion data, lidar data, and metadata on driving conditions. In the experiments, only blurred images were used.

Each image is associated with GNSS/IMU data that provides reliable navigation and positioning information. We shall focus on the task of vehicle trajectory planning and leverage the positioning information to automatically anotate the image frames as in [33]. The ground truth is constructed by interpolating 17 points from the GNSS/IMU data, 3D-points in the trajectory from the original position of the car. The target distances of the 17 points are given by $\{t_i\}_{i=1}^{17}$ where $t_i = 5i$ for $1 \leq i \leq 8$, $t_i = 10(i-8)+40$ for $i \leq 12$, and $t_i = 15i(i-12)+80$ $i > 12$. Hence, the annotations emphasizes accuracy in the predicted trajectory close to the ego vehicle.

The dataset is split into a training, test and a server defense set, as seen in Fig. 1. To facilitate federated learning, the training set is further divided into separate sets for each global round and client. This partitioning is different from vanilla federated learning where the dataset remains static at each client. In self driving, however, the car may be unable to store the data locally and must, hence, discard some of the data to make room for new. We capture this behavior by replacing the local data of all clients in every training round. The test set is used to evaluate the model after each round. For the OTA, a test set was also created by including the backdoor trigger pattern in each image, leaving the ground-truth trajectory unchanged, to assess the attack success. Finally the server defense set may be used in conjunction with mitigation strategies employed by the server during training.

B. Vehicle Trajectory Prediction

We employ the MobileNet-V3 [34] as the backbone of the trajectory prediction, pretrained on the ImageNet dataset [35]. MobileNet-V3 is a convolutional neural network optimized for mobile phone CPUs. We replace the head of network by 3 linear layers: 1024 neurons with ReLU activation, 512 neurons with ReLU activation, and 51 neurons without activation function. The 51 neurons in the final layer correspond to the 17 three-dimensional points $\{\hat{p}_i\}_{i=1}^{17}$, $\hat{p}_i \in \mathbb{R}^3$, representing the predicted trajectory. To facilitate the learning, we let $\hat{p}_{ij} \in [0,1]$, $j \in [3]$, and multiply $\hat{p}_{ij}$ with t_i, to obtain the point's position relative to the vehicle. This allows the network to treat each predicted point equally.

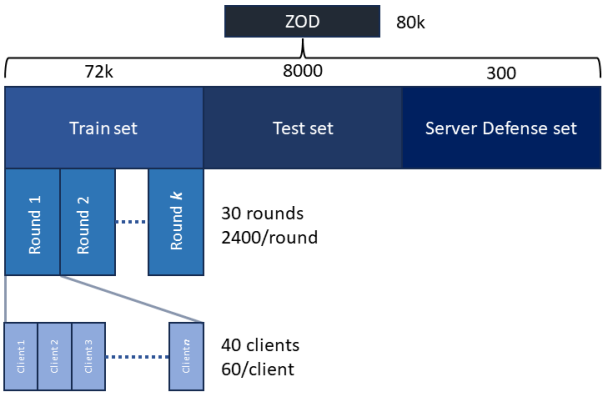


Fig. 1: Visual representation of the dataset split, illustrating the number images of the ZOD-dataset that were used for training (and how they are partitioned among clients), testing, and server defense.

During training, we employ the Adam optimizer with a learning rate of 0.001, a batch size of 32, and the L1-loss function. Hence, for a given data point, consisting of an image x and a ground-truth trajectory $\{p_i\}_{i=1}^{17}$, the loss is obtained as

$$\ell(x, \{p_i\}_{i=1}^{17}; \theta) = \frac{1}{17}\sum_{i=1}^{17} \|p_i - \hat{p}_i\|_1 \tag{4}$$

where $\{\hat{p}_i\}_{i=1}^{17} = \theta(x)$ is the predicted trajectory.

C. Federated Learning

For the federated learning, we consider a network consisting of 40 clients. The training is performed over 30 global training rounds where each round consists of 3 local epochs. As already mentioned, the clients are assumed to have collected a new dataset in the beginning of each training round. This is illustrated in Fig. 1 where the 72K training samples are split over the 30 training rounds and then, within each training round, further split over the 40 clients resulting in 60 data points per client. Note that the data partitioning is performed randomly. Although a random data partitioning is not realistic, e.g., consecutive data frames have a strong correlation in environment and weather, such partitioning was not feasible at the time of writing and left as an interesting future direction of study.

We assume that 4 out of the 40 clients are malicious. Furthermore, during the federation, the server randomly samples 10 out of the 40 clients in each round. Hence, the prevalence of malicious users may vary between 0% to 40% in a given training round. The aggregation at the server is achieved by federated averaging [3]. Pseudo code for the federated learning procedure is provided in Algorithm 1.

V. Experiments

In this section, we assess the robustness of FL using various poisoning attacks and defense strategies. The experiments were performed on a single NVIDIA Quadro

TABLE I: Training score of the resulting model in conjunction with a given attack-defense combination.

Attack Name	No-Defense	KRUM	MULTI-KRUM	LFR	FLTRUST	PCA DEFENSE	LOSS DEFENSE	TRIMMED MEAN	LOSSFUSION
No Attack	3.114	3.564	3.460	3.057	3.043	3.260	3.027	3.158	2.990
Label-flipping	7.924	3.381	3.446	3.058	3.616	3.397	3.071	4.015	3.043
GRADIENT ASCENT	250.489	3.518	3.450	3.994	3.773	4.737	3.102	7.552	3.030
MSA	4.402	3.456	3.447	3.067	3.130	3.190	4.437	3.178	3.013
FLSTEALTH	$34.23 \cdot 10^{10}$	5.423	4.685	42.478	483.935	$21.91 \cdot 10^{8}$	3.025	$32.63 \cdot 10^{5}$	3.086

Algorithm 1 Federated Learning Procedure

Server side
$\theta_{\text{global}} \leftarrow$ pretrained MobileNetV3
for r from 1 to 30 **do**
  $S_r \leftarrow$ 10 clients selected at random
  Broadcast θ_{global} to S_r
  for client $c \in S_r$ **do**
    $\theta_c \leftarrow$ TrainClient(θ_{global}, r)
  end for
  $\theta_{\text{global}} \leftarrow$ Aggregate(θ_{global}, $\{\theta_c\}_{c \in S_r}$)
end for

Client side
function TRAINCLIENT(θ_{global}, r)
  $D_r \leftarrow$ get dataset for current client and round
  $\theta_{\text{client}} \leftarrow \theta_{\text{global}}$
  for each epoch e from 1 to 3 **do**
    for each batch $b \in D_r$ **do**
      Update θ_{client} using b
    end for
  end for
  return θ_{client}
end function

RTX5000 GPU with 8 cores, 40GB RAM and 500GB disk space. The duration of one experiment on the entire dataset is 20-30 minutes.

A. Untargeted Attacks

To measure the outcome from the federated training, the test loss of the global model is averaged over the last 10 training rounds, we refer to this metric as *training score*. A high training score indicates a global model with poor performance, potentially due to a successful attack. On the other hand, a good model yields a low training score, possibly due to a weak attack or of a successful defense. Moreover, we report the training scores as the average over 10 separate runs, i.e., each (attack, defense) combination is executed 10 times.

We consider 5 different poisoning attacks, including our novel FLSTEALTH attack, and 8 different mitigation strategies. As a baseline, we also provide the result without any mitigation strategies referred to as No-Defense. For attacks requiring parameters, we consider: 1) in the label flipping ground truth trajectories are multiplied by -100, 2) for MSA, we shuffle 100 random rows in the weight matrix of each linear layer, 3) for FLSTEALTH, the byzantine model is trained for 15 epochs using a learning rate of 0.0001 and $\kappa = 10^{-9}$. Note that a small value of κ is typically required as the mean-squared error between the honest and byzantine models is in general much smaller than the loss. Similarly, for defenses requiring parameters, we use: 1) in Krum, we use 4 byzantine clients, 2) in Multi-Krum, we use 4 byzantine clients and 6 models to be aggregated, 3) in Trimmed Mean, after ordering the client updates based on magnitude, two clients are removed from the bottom and from the top of the ordering, and 4) for PCA DEFENSE, LFR, LOSS DEFENSE, and LOSSFUSION, 4 clients are excluded in each round. Note that the parameters are chosen in favor of the defenses as the correct number of malicious clients from the entire client set is used.

The LOSSFUSION defense mehchanism is a simple fusion of LFR and LOSS DEFENSE after running them separately. In particular, let θ_{LFR} and θ_{LD} denote the resulting model parameters after employing the two defense mechanisms separately. Then, LOSSFUSION selects the model parameters as

$$\theta_{\text{LF}} = \begin{cases} \theta_{\text{LFR}} & \text{for } \ell(D_{\text{server}}; \theta_{\text{LFR}}) < \ell(D_{\text{server}}; \theta_{\text{LD}}) \\ \theta_{\text{LD}} & \text{otherwise} \end{cases}$$

where $\ell(D_{\text{server}}; \theta)$ is the average loss on the server's defense dataset using a model θ. LOSSFUSION aims at alleviating the weakness of only considering pre-aggregation losses in LOSS DEFENSE and of only looking at post-aggregated losses in LFR. Hence, LOSSFUSION effectively eliminates attacks targeting either LFR or LOSS DEFENSE since now both defenses must be bypassed.

In Table I, we illustrate the average training score for each attack-defense combination. It can be seen that some combinations, particularly involving FLSTEALTH, results in very high training scores. The reason for this is that some of the attacks can be made arbitrary strong when able to bypass the defense. Among the attacks, FLSTEALTH achieves the largest training score for all defenses but the LOSSDEFENSE. On the other hand, among the defenses, LOSSFUSION achieves the lowest training score on all attacks but FLSTEALTH.

B. Targeted Attacks

The design of our targeted attack, OTA, involves three steps: 1) how to inject a trigger to an image, 2) how to alter the ground truth trajectory, and 3) decide how large portion of the data to poison.

1) Trigger Injection: Although there are many ways to design a trigger, in this paper, a simple square pattern was chosen. Based on this choice, multiple features were studied, e.g., size, color, and total number of squares added.

Empirically, position and size surfaced as the main factors for a successful attack; varying the color of the square between red, green and white, or increasing the number of squares did not affect the overall performance of OTA. Hence, for simplicity, only one red square were used for the final experiments.

To understand the impact of the square's position, experiments were conducted positioning it at the top-left corner, the center of the image, or at a random position for each image in the byzantine dataset. From these experiments, random position often went unnoticed by the defenses and hence that option was used for further experiments. However, we remark that positioning the square in the center performed the best but was deemed unrealistic, see Section VI-B).

Finally, the size of the square only matters when it gets too small for the network to notice. The size was set as a percentage of the height of the image and performance dropped at around 5% of the height. Sizes of up to 16% of the image height was used with success, and for consistency in further experimentation a size of 10% was used.

2) Altering the Ground-Truth Trajectory: When a trigger is injected to a data sample, the corresponding ground-truth trajectory should also be modified in order to change the behavior of the model. We considered three such modifications: 1) make the car turn by the end of its path, 2) make the car go straight, and 3) make the car sig-sag around the ground-truth trajectory. From experimenting, the attack was deemed successful only when the car was made to turn, hence, for the final experiments, a trigger will force the car to turn.

It should be noted that a turn change can be achieved in several ways, e.g., by changing the angle of the turn, the sharpness of the turn, or the direction (left/right). As most variations demonstrated similar result, a set-up with a turn to the right by modifying the last 5 points of the ground truth was chosen.

3) Number of Poisoned Examples: The final component of the OTA is to choose the amount of data samples to poison. From experiments with 20% to 100% of the data samples being poisoned, a trade-off was identified. A too large portion resulted in the backdoor becoming ineffective as the trigger is mostly present resulting in the entire dataset being poisoned and, consequently, the client model being easily identified as malicious. On the other hand, a small portion of poisoned data resulted in the model not learning the trigger at all. Empirically, we found that a portion of 30% of the dataset being poisoned yielded good results. In Fig. 2 the loss trajectories are illustrated for a successful targeted attack. From the test loss trajectory on the backdoor test set, see Section IV-A, we notice that the loss trajectory increases by the end of the learning procedure which indicates a successful attack, i.e., the predicted trajectory deviates from the ground-truth trajectory in the presence of a trigger. Another way of visualizing a successful backdoor attack is by the attention heat maps, as shown in Fig. 3. The series of images shows how the attention of the model is shifted from the road to the top left corner after the attack.

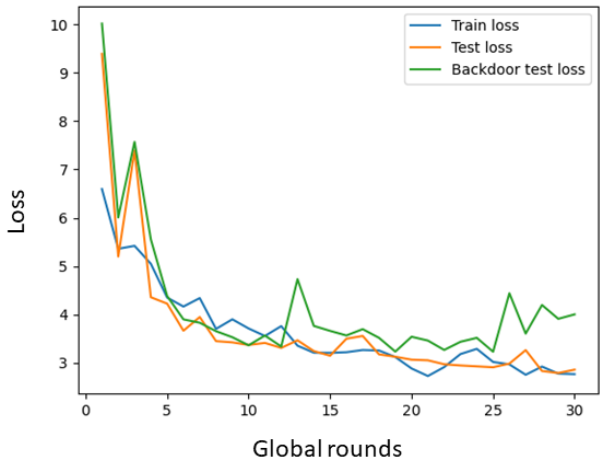


Fig. 2: Loss trajectories for a successful targeted attack (OTA) against LFR defense.

TABLE II: Training and backdoor score from an OTA.

Defense	Training score	Backdoor score	Difference
No defense	3.19	3.52	0.33
LFR	2.92	3.31	0.33
LOSS DEFENSE	2.99	3.18	0.19
PCA DEFENSE	3.27	3.48	0.21
MULTI-KRUM	3.24	3.30	0.06
FLTRUST	3.01	3.40	0.39
LOSSFUSION	3.08	3.28	0.20

4) Results: To measure the success of OTA, we consider both the training score, similar to untargeted attacks, but also a metric called backdoor score, computed similarly to the training score but over the backdoor test dataset, i.e., the same test set as in the training score but with triggers injected in images. We expect a successful OTA to achieve a low training score, i.e., perform well on images without triggers, while simultaneously achieving a large backdoor score, i.e., deviate from ground-truth trajectories when triggers are present. Table II illustrates the performance of OTA against six defense mechanisms with parameters chosen as in Section V-A. The difference between the training score and the backdoor score indicates the effectiveness of the attack with a larger difference yielding a more successful attack. The values in each row in Table II is the average over 5 independent runs.

From Table II, it can be seen that LOSS DEFENSE and LOSSFUSION are effective at mitigating OTA, yielding a difference of 0.19 and 0.20, respectively. Although MULTI-KRUM displays the lowest difference of 0.06, the training score is large. Visual inspection of predictions obtained from models trained with MULTI-KRUM mitigation also entails that the model is poisoned, i.e., predictions follow the expected behavior when exposed to the trigger.

To further test the robustness of OTA, an attack was performed in a more realistic setting, as shown in Fig. 4.

(a) Attention heatmap of global model on a normal image.

(b) Trigger injected (a red square) in the top left corner.

(c) Attention heatmap of global model after backdoor is added to the image.

Fig. 3: Change of model attention when a backdoor is added to the picture (Frame #074220 in the ZOD-dataset)

Fig. 4b displays a road with a person showing a trigger pattern on a computer screen. A photo without the person was then generated, using image processing tools, in order to keep the environment fixed. The model, subject to the OTA, employing the LOSS FUSION defense was then used to predict the trajectory on each image. Without the trigger pattern the model produces a reasonable prediction of the trajectory, see Fig. 4a, and when the backdoor pattern was introduced, the model sends the car to the right, see Fig. 4b, which, in this case, is the opposite of the intended direction.

VI. DISCUSSION

In the following section, we discuss our results pertaining to untargeted and targeted attacks, respectively.

A. Untargeted Attacks

The FLSTEALTH attack aims to deteriorate the global model while remaining undetected. This proves to be effective against all considered defenses, apart from LOSS DEFENSE and LOSSFUSION. For KRUM, MULTI-KRUM, FLTRUST and PCA DEFENSE, these results are expected as they rely on a similarity score for each client and mitigate the impact of dissimilar clients. Since FLSTEALTH is designed to provide poisoned models similar to those of honest clients, the malicious clients will have a similar similarity score to an honest client. FLSTEALTH is also expected not to bypass LOSS DEFENSE as it is designed to increase the loss which is exactly the signal that LOSS DEFENSE operates on. Decreasing κ will improve the chances of bypassing also LOSS DEFENSE but will also reduce the effect of the attack.

Interestingly, FLSTEALTH and the related GRADIENT ASCENT attack both perform well against LFR. We observe that this is because the attack sometimes, but rarely, bypasses LFR completely. For each such instance, at least two attackers are present and removing one of them results in a worsened model. This counter intuitive phenomenon is due to the inner workings of LFR that removes clients sequentially based on the loss impact of each client. When multiple attackers are present, their updates may partially cancel out and may, in some cases, result in a low loss when both are included but an increased loss when one is removed. Since LFR does not take into account the relationship between multiple clients, the defense will not realise that the best strategy is to remove both clients but will, instead, remove 4 other clients, amplifying the attack further since it now contributes more to the averaged model.

B. Targeted Attacks

OTA successfully evades all the defenses, hence poisoning the global model and injecting the trigger into all vehicles in the federation. Since the model is trained to make good predictions when no trigger pattern is present the targeted model will have low loss. This is the reason why loss-based mitigation strategies are unsuccessful. The second category of defenses focus on the similarity of the received client gradients. However, as the malicious clients only poison 30% of their local data, their updates will be similar to that of a benign client, rendering similarity-based defenses ineffective.

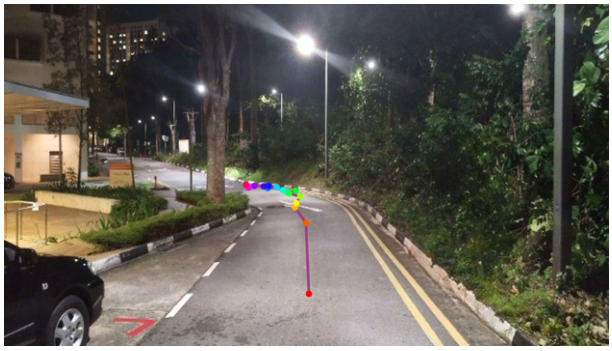
During the experiments, some defenses were sometimes able to counter or cancel out OTA in a single training round. However, if the malicious client manages to bypass the defense in only a single round, the trigger will be present for all clients going forward. This may further allow the attacker to bypass the defense in future rounds, amplifying the effect of the attack.

As mentioned in Section V-B, there are several ways of adding trigger patterns. The empirical results suggested that the best positioning for a trigger pattern is in the center of the image. This is expected since that square would cover the most important part of the image, where the model's attention is focused, i.e., the road. However, in real life this would limit the position of the attacker and make the attack more difficult to execute, hence, this positioning was rejected.

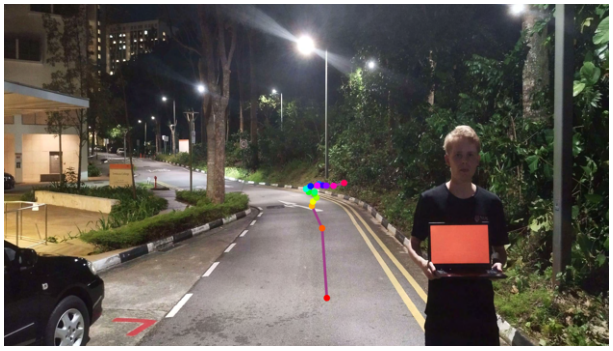
VII. CONCLUSION

This paper studies vulnerabilities of federated learning applied in the area of regression tasks within autonomous driving. We have introduced two novel attacks: 1) an untargeted attack called FLSTEALTH tailored to deteriorate the global model while remaining stealthy and 2) a targeted attack OTA aiming to inject triggers to make the car turn when exposed to the trigger. A thorough assessment of the attack success was performed by comparing to other types of attacks and to common poisoning mitigation strategies in federated learning.

Our results have highlighted the significant threat posed by backdoor attacks, calling for effective detection methods and exploring ensemble techniques that combine different approaches that could enhance defenses against targeted



(a) Trajectory prediction on road in Singapore



(b) The same image as Fig. 4a but with a malicious actor showing the trigger pattern.

Fig. 4: OTA performed in practice.

attacks. Notably, we observed that none of the existing defenses effectively countered OTA. Finally, we demonstrated the benign effects of combining multiple defensive strategies, as demonstrated by the introduced LOSSFUSION defense.

REFERENCES

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv:1604.07316*, 2016.

[2] European Union, "General data protection regulation (GDPR) information portal," 2023. Available at: `https://gdpr-info.eu`.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.

[4] M. Aparna, R. Gandhiraj, and M. Panda, "Steering angle prediction for autonomous driving using federated learning: the impact of vehicle-to-everything communication," in *IEEE Int. Conf. on Comp. Comm. and Netw. Technologies (ICCCNT)*, 2021.

[5] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, "Opportunities of federated learning in connected, cooperative, and automated industrial systems," *IEEE Communications Magazine*, vol. 59, no. 2, 2021.

[6] A. Nguyen, T. Do, M. Tran, B. X. Nguyen, C. Duong, T. Phan, E. Tjiputra, and Q. D. Tran, "Deep federated learning for autonomous driving," in *IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[7] Y. Chen, C. Wang, and B. Kim, "Federated learning with infrastructure resource limitations in vehicular object detection," in *IEEE/ACM Symposium on Edge Computing (SEC)*, 2021.

[8] D. S., K. N., and S. Athavale, "Turn signal prediction: A federated learning case study," *arXiv 2012.12401*, 2020.

[9] D. Deveaux, T. Higuchi, S. Uçar, C.-H. Wang, J. Härri, and O. Altintas, "On the orchestration of federated learning through vehicular knowledge networking," in *IEEE Vehicular Networking Conference (VNC)*, 2020.

[10] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Neurips*, vol. 30, 2017.

[11] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLtrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv 2012.13995*, 2020.

[12] A. Huang, "Dynamic backdoor attacks against federated learning," *arXiv 2011.07429*, 2020.

[13] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, "Data poisoning attacks on federated machine learning," *IEEE Internet of Things Journal*, vol. 9, no. 13, 2021.

[14] X. Li, G. Kesidis, D. J. Miller, and V. Lucic, "Backdoor attack and defense for deep regression," *arXiv:2109.02381*, 2021.

[15] S. Wang, Q. Li, Z. Cui, J. Hou, and C. Huang, "Bandit-based data poisoning attack against federated learning for autonomous driving models," *Expert Systems with Applications*, vol. 227, 2023.

[16] M. Alibeigi, W. Ljungbergh, A. Tonderski, G. Hess, A. Lilja, C. Lindström, D. Motorniuk, J. Fu, J. Widahl, and C. Petersson, "Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving," in *IEEE/CVF International Conference on Computer Vision*, 2023.

[17] S. Mahloujifar, M. Mahmoody, and A. Mohammed, "Universal multi-party poisoning attacks," in *ICML*, 2019.

[18] R. Guerraoui, S. Rouault, *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *ICML*, 2018.

[19] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *ICLR*, 2019.

[20] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *AISTATS*, 2020.

[21] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *AISec*, 2017.

[22] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *ESORICS*, 2020.

[23] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," in *IEEE Symposium on Security and Privacy (SP)*, 2022.

[24] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *Neurips*, vol. 32, 2019.

[25] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to {Byzantine-Robust} federated learning," in *USENIX*, 2020.

[26] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Network and Distributed Systems Security Symposium*, 2021.

[27] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *arXiv:1206.6389*, 2012.

[28] M. Yang, H. Cheng, F. Chen, X. Liu, M. Wang, and X. Li, "Model poisoning attack in differential privacy-based federated learning," *Information Sciences*, vol. 630, 2023.

[29] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, 2019.

[30] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *ICML*, 2018.

[31] V. Valadi, X. Qiu, P. P. B. de Gusmão, N. D. Lane, and M. Alibeigi, "Fedval: Different good or different bad in federated learning," *USENIX*, 2023.

[32] M. Xhemrishi, J. Östman, A. Wachter-Zeh, and A. G. i Amat, "FedGT: Identification of malicious clients in federated learning with secure aggregation," *arXiv:2305.05506*, 2023.

[33] A. Viala Bellander and Y. Ghafir, "Towards federated fleet learning leveraging unannotated data," Master's thesis, Chalmers University of Technology, 2023.

[34] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, "Searching for mobilenetv3," in *IEEE/CVF International Conference on Computer Vision*, 2019.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Ima-

genet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.