SecFedDrive: Securing Federated Learning for Autonomous Driving Against Backdoor Attacks

Rahul Kumar*, Gabrielle Ebbrecht*, Junaid Farooq†, Wenqi Wei*, Ying Mao*, and Juntao Chen*

2024 IEEE Conference on Communications and Network Security (CNS) | 979-8-3503-7596-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/CNS62487.2024.10735501

***Abstract*—Federated learning (FL) enables collaborative model training without sharing the private data of each individual participant, making it well-suited for autonomous driving applications. Preserving the integrity of sensitive data is crucial for the security of these systems due to the direct implications for passenger safety. Although federated model training enhances data privacy for individual participants, it remains vulnerable to stealthy backdoor attacks that can alter the global model, potentially threatening system reliability in high-stakes scenarios such as autonomous driving. We introduce SecFedDrive, a robust defense mechanism against backdoor attacks on a decentralized deep FL system. Our approach reduces attack success rates to as low as 2%, significantly enhancing the security and reliability of FL models in autonomous driving environments, and demonstrates significant reductions in training time relative to a comparable architecture.**

I. Introduction

Deep learning, a powerful technique for extracting complex data patterns, is a transformative technology enabling breakthroughs across domains including natural language processing and computer vision. Autonomous vehicles, which are revolutionizing transportation with a promise of enhancing vehicles' safety, efficiency, and accessibility, collect and process vast amounts of data from individual vehicles to train a global model. Traditional approaches to training deep learning models are centralized, processing all participants' training data on one central server and training the model on one consolidated dataset; however, this raises concerns about the security of sensitive participant data and personal information.

A. Federated Learning for Autonomous Driving

Federated learning (FL) [1] is an emerging paradigm in machine learning that mitigates threats to data privacy and sovereignty via collaborative learning, where multiple devices or servers train their respective data separately, aggregating their local models to comprise a global model without data exchange. FL offers additional benefits, e.g., enhanced efficiency as only parameter updates are communicated, and scalability by leveraging the computational resources of many devices.

FL is highly relevant to autonomous driving [2] as the sensor of each vehicle continuously captures vast amounts of sensitive data, including image, video, and location information. FL ensures the privacy of sensitive data while taking advantage of its heterogeneity—such as different driving conditions, environmental characteristics, road infrastructure, etc.—to better generalize global models and enable continuous model updates to enhance system performance and safety. With the increasing prevalence of autonomous vehicles, FL offers an efficient solution that reduces overhead for centralized storage and processing. However, despite its advantages, this paradigm does not eliminate all threats to data and model security.

B. Backdoor Attacks on Federated Models

An adversary may launch a black-box backdoor attack on a federated learning system via data poisoning by stealthily injecting malicious data into the training set [3], often at the level of individual pixels that induce subtle perturbations to mislead the model during training while going undetected. The adversary may carry out such an attack across multiple nodes to further skew the model. Model poisoning [4], the white-box alternative, compromises the integrity of the learning process by strategically manipulating parameters or the objective function to bias the model toward specific classes. Distributed backdoor attacks allow adversaries to exploit the communication protocol or aggregation processes of FL to inject poisoned updates and compromise the global model by attacking participants [5].

Due to the privacy-preserving aggregation of participants' local models, anomalies are difficult to detect by the centralized server [6]. As such, innovative defense mechanisms have been proposed to combat the growing landscape of threats to FL. Neural Cleanse [7] is one such strategy designed to thwart backdoor attacks by neutralizing malicious triggers embedded in FL models. This scheme probes the internal structure of neural networks for anomalous patterns indicative of backdoor triggers. Advanced detection algorithms and analysis techniques enable it to proactively cleanse FL models of hidden vulnerabilities, offering a vital layer of protection against backdoor attacks in an increasingly adversarial environment.

The contributions of this paper is summarized as follows.

- We develop SecFedDrive, a secure FL architecture for defending against strategic backdoor attacks in autonomous driving. This defense mechanism incorporates two security layers in the learning architecture, Residual Check and Neural Cleanse, that improve the trustworthiness of vision-based autonomous driving decision-making.

*The authors are with the Department of Computer and Information Sciences, Fordham University, New York, NY, 10023 USA. E-mail: {rkumar42, gebbrecht, wwei23, ymao41, jchen504}@fordham.edu

†The author is the Department of Electrical & Computer Engineering, College of Engineering and Computer Science, University of Michigan-Dearborn, Dearborn, MI 48128 USA. E-mail: mjfarooq@umich.edu

This work is partially supported by Grant ECCS-2138956 from the National Science Foundation and a grant from the Fordham Office of Research.

979-8-3503-7596-1/24/$31.00 ©2024 IEEE

- We evaluate our approach extensively on two public datasets, with results showing significant reductions in attack success rate when compared with the FL architecture lacking the proposed security measures.
- Our proposed FL architecture is lightweight in terms of training time, reducing its vulnerability to backdoor attacks through image poisoning during training.

II. Related Work

There are a variety of existing works across the fields of federated learning, backdoor attacks, and related countermeasures. Our work spans all of these focus areas to uncover the impact of malicious participants injecting backdoor triggers on an FL network, and the extent to which these attacks can be mitigated.

A. Federated Learning

McMahan et al. propose a decentralized FL scheme for training deep neural networks by aggregating locally computed updates, which is robust to non-IID data distributions and enhances privacy while maintaining performance [1]. Nguyen et al. propose FADNet [2], a communication-efficient peer-to-peer decentralized deep federated learning (DFL) framework for training autonomous driving models, which performs competitively with centralized alternatives and state-of-the-art methods of processing autonomous driving datasets. It is used as a baseline with which to compare our secure FL network.

B. Backdoor Attacks

Tu et al. investigate the vulnerabilities and robustness of collaborative multi-agent systems against adversarial perturbations and complexities of executing black-box transfer attacks, proposing methods for generating adversarial perturbations that are indistinguishable yet detrimental to the system's functionality [8]. Bagdasaryan et al. proposes a "blind" method for backdoor injection into deep learning models by compromising loss-value computation during training without access to training data or the model [9] and introduce an FL-specific model-poisoning backdoor attack that is trained to evade detection with a train-and-scale technique [6]. The distributed attack model presented in [5] also exploits the characteristics of FL by decomposing a global trigger to embed it locally on a set of participants.

C. Countermeasures

Wang et al. propose Neural Cleanse [7], the first robust and generalizable system for detecting and mitigating backdoor attacks in deep neural networks (DNNs) with proven effectiveness when deployed against backdoor variations. This work is foundational to the architecture of our proposed defense mechanism. Alternative countermeasures include robustness certification based on model smoothing [10] and post-training model pruning [11].

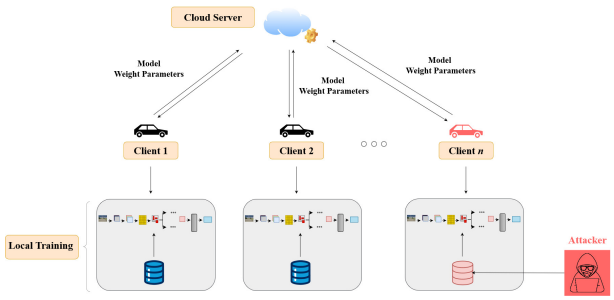

Fig. 1: Overview of federated learning: Local weights are sent to the cloud server and the aggregated parameters are sent back. An attacker may target client image data to report false parameters and have malicious effects on the global model, impacting vehicles' steering policy.

III. Problem Formulation

We first introduce mathematical foundations for our federated learning case and the backdoor attack model used to maliciously alter the learning model's behavior. The federated network is comprised of N clients communicating with a central server S to train the vehicle steering policy. Each client C_i, $i \in \{1,...,N\}$, trains a deep learning model on its local dataset and sends the learned weights θ_i to S. S then generates a global policy for predicting the steering angle from the calculated aggregated weights θ. Fig. 1 shows the described architecture.

A. Federated Learning Architecture

Federated learning can be described as an iterative process of local optimization and global aggregation [12], converging towards a model that approximates the minimization of a global loss function. Consider a machine learning problem with the goal of minimizing a global objective function $F(\mathbf{w})$ over a model parameter vector $\mathbf{w}$. This global objective function is the average of local objective functions $F_k(\mathbf{w})$ from K different clients, described by the following equation:

$$F(\mathbf{w}) = \frac{1}{K}\sum_{k=1}^{K} F_k(\mathbf{w}), \tag{1}$$

where $F_k(\mathbf{w})$ represents the objective function for the k-th client. Each client k has its own local dataset $\mathscr{D}_k$, and the local objective function is typically defined as:

$$F_k(\mathbf{w}) = \frac{1}{|\mathscr{D}_k|}\sum_{i\in\mathscr{D}_k} \ell(\mathbf{w};\mathbf{x}_i,y_i), \tag{2}$$

where $\ell(\mathbf{w};\mathbf{x}_i,y_i)$ is the loss function for the model parameter $\mathbf{w}$ on data point $(\mathbf{x}_i,y_i)$. The central server initializes the global model parameter $\mathbf{w}_0$.

Federated learning proceeds in a series of communication rounds indexed by $t = 0,1,2,\ldots,T$. At the beginning of round t, the central server transmits the latest global model parameter $\mathbf{w}_t$ to all clients, who then update their parameters based on a local dataset via local optimization, i.e., as gradient descent:

$$\mathbf{w}_t^{(k)} \leftarrow \mathbf{w}_t - \eta\nabla F_k(\mathbf{w}_t), \tag{3}$$

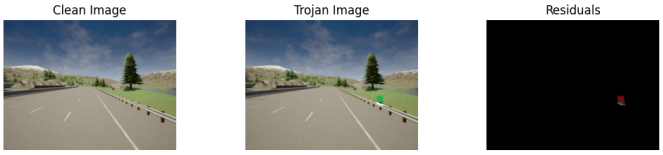

Fig. 2: A backdoor trigger pattern demonstration. The Trojan image contains a small green patch beneath the tree as the trigger. The residual highlights the difference between images.

where η is the learning rate and $\mathbf{w}_t^{(k)}$ is the updated model parameter for client k after local training. Clients send their updated local model parameters $\mathbf{w}_t^{(k)}$ back to the central server which aggregates them. The aggregation is a simple average or a weighted average if clients have different data sizes:

$$\mathbf{w}_{t+1} = \frac{1}{\sum_{k=1}^{K} |\mathscr{D}_k|} \sum_{k=1}^{K} |\mathscr{D}_k| \mathbf{w}_t^{(k)}. \tag{4}$$

This process repeats for a predetermined number of communication rounds T or until convergence criteria are met, with the goal of minimizing the global objective function $F(\mathbf{w})$ [1].

B. Backdoor Attacks

The FL framework faces the vulnerability of strategic backdoor attacks, as illustrated in Fig. 1. Let us define the trigger injection as the following:

$$A(x, m, \delta) = x', \tag{5}$$

$$x'_{i,j,c} = (1 - m_{i,j}) \cdot x_{i,j,e} + m_{i,j} \cdot \delta_{i,j,c}, \tag{6}$$

where $A(\cdot)$ denotes the function applying a trigger to the original image x. δ is a 3D matrix of pixel color intensities with identical height, width, and color channel dimensions to the input image. m is a 2D matrix that serves as a mask over the images, hence determining the extent to which the trigger can replace the original image. Here, we investigate a 2D mask that has been applied to all color channels of the pixel. The values in the mask range from 0 to 1. When $m_{i,j} = 1$ for a specific pixel (i, j), the trigger overwrites the original color ($x'_{i,j,c} = \delta_{i,j,c}$). When $m_{i,j} = 0$, the original color is not modified ($x'_{i,j,c} = x_{i,j,c}$) [7]. The mask we use in our experiments consists of the right side of the image having some pixels that are disguised, as shown in Fig. 2.

IV. SecFedDrive Architecture

Backdoor data poisoning attacks can be executed through various attack vectors, such as a spoofed agent, a cyberattacker targeting the data during transmission, or a malicious insider manipulating the training data. Due to the high risk posed by attacks on autonomous driving systems, adversarial behavior should be anticipated, and proactive security measures should be implemented against it. We develop SecFedDrive, an in-depth defense mechanism comprised of two security enhancement layers in the learning architecture: Residual Check and Neural Cleanse. Specifically, the residual defense prevents clients from learning malicious data by identifying backdoors before training, with Neural Cleanse's neuron pruning and unlearning methods serving as an additional defense mechanism.

A. Multilayer Defense Mechanisms

1) Residual Check: A preliminary residual defense blocks images with pixel-based triggers from entering the model training phase. In image processing, residuals are the absolute differences between corresponding pixel intensities in two grayscale images. For a given image, the previous image passed into the model is compared with it. The images are first converted from RBG to grayscale, then the residual $R(x,y)$ at a given pixel coordinate (x,y) is calculated as the absolute value of the difference between intensity values $L_1(x,y)$ and $L_2(x,y)$ of the two grayscaled images, denoted as the following:

$$R(x,y) = |L_1(x,y) - L_2(x,y)|. \tag{7}$$

Analyzing residuals is an effective technique for capturing modifications to images [13], and any unusual differences between images can indicate backdoor triggers. However, these triggers may not be evident if the differences between images are too large. For example, a squirrel running across the road and suddenly appearing in the current image while not being present in the previous image may be flagged as a trojan attack.

2) Neural Cleanse: Neural Cleanse determines the minimum perturbation δ required to modify any input x so that the model f_θ classifies $x+\delta$ into a target class t. Mathematically, this is formulated as an optimization problem:

$$\delta_i^* = \arg\min_{\delta} \|\delta\|_p \quad \text{s.t.} \quad f_{\theta_i}(x+\delta) = t, \quad \text{for } x \in D_i. \tag{8}$$

In this context, $\|\cdot\|_p$ represents the ℓ_p-norm, commonly the ℓ_1- or ℓ_2-norm. This optimization can be solved using gradient-based methods such as Projected Gradient Descent (PGD).

In a federated learning scenario, the server receives δ_i^* from each client i. The aggregated perturbation $\bar{\delta}$ is computed as:

$$\bar{\delta} = \frac{1}{N} \sum_{i=1}^{N} \delta_i^*. \tag{9}$$

Neural Cleanse employs an anomaly detection mechanism to identify backdoor attacks. The intuition is that a backdoored client will yield a significantly smaller $\|\delta_i^*\|_p$ compared to non-backdoored clients. Let μ and σ denote the mean and standard deviation of the perturbation norms $\|\delta_i^*\|_p$ across all clients. The anomaly score α_i for each client is given by:

$$\alpha_i = \frac{\|\delta_i^*\|_p - \mu}{\sigma}. \tag{10}$$

A high anomaly score α_i indicates a potential backdoor attack. By setting a threshold τ, we can flag and isolate suspicious clients so that if $\alpha_i > \tau$, then client i is suspected of being backdoored.

The objectives of the optimization are as follows. To analyze a given target label y, a trigger (m, δ) that would misclassify clean images as $\eta(y)$ must first be found. Subsequently, a

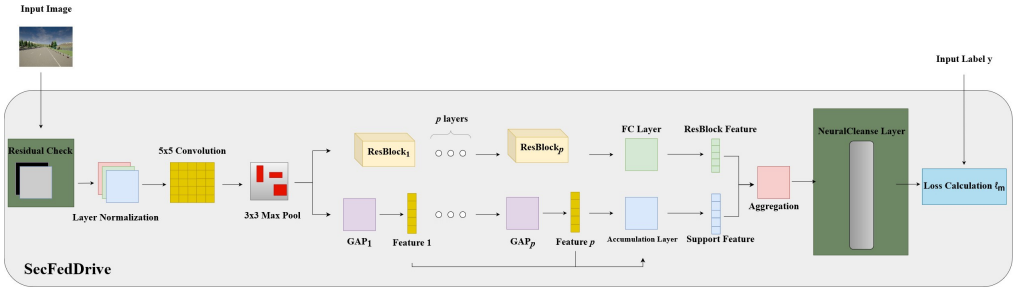

Fig. 3: Visualization of the SecFedDrive architecture, composed of a series of convolutional layers and residual blocks, designed to process input data for autonomous vehicles in a federated learning setting. Security measures, Residual Check and Neural Cleanse, are implemented at the boundaries of the neural network architecture, safeguarding both the data entering the input layer and the loss calculated from the aggregated weights.

trigger that modifies a limited portion of the image must be found. The formulation of such objectives is described as:

$$\min_{m,\delta} \quad \ell(y_t, f(A(x,m,\delta))) + |m| \\ \text{for } x \in X, \tag{11}$$

where $\ell(\cdot)$ is the loss function quantifying classification error—cross entropy in our experiment—and $f(\cdot)$ is the prediction function. The optimization problem is solved using a set of clean images X from the Udacity and CARLA datasets.

B. Main Architecture of SecFedDrive

The architecture of the proposed model, pictured in Figure 3, is implemented using the PyTorch deep learning framework and inherits from the nn.Module class. The network begins with a convolutional layer (32 output channels, kernel size 5, stride 4), followed by a max pooling layer (kernel size 3, stride 2). Three residual blocks follow, with each consisting of a sequence of operations including batch normalization, ReLU activation, and convolutional layers, progressively increasing output channels from 32 to 128. Following the residual blocks, additional convolutional layers with varying output channels and strides reduce the spatial dimensions of the input while increasing the number of channels.

The model incorporates a support feature mechanism, wherein the output of specific convolutional layers is reshaped and subjected to a global average pooling operation to obtain support features. These support features are accumulated using a fully connected layer to produce a single feature. A fully connected layer with an input size of 7 and an output size of 7 is included, along with a dropout layer with a dropout probability of 0.5 and a ReLU activation function.

The training process is as follows: In the forward propagation phase, the input is passed through the initial convolutional layer and max pooling layer. The output of the max pooling layer is then processed by the first residual block, and the result is added to the output of the convolutional layer applied to the output of the max pooling layer. After passing through the residual blocks and convolutional layers, the output is flattened and passed through the fully connected layer, followed by ReLU activation and dropout. The final output is obtained by multiplying the flattened output with the accumulated support feature and taking the mean across the appropriate dimension.

C. Security Layers of SecFedDrive

The Residual Check, inserted at the input layer of the architecture as seen in Figure 3, captures the residual between subsequent frames in an autonomous vehicle feed using OpenCV. In the context of our experiments, the comparison between the frames was conducted on all the input data before entering the model. This pre-check flags images with unusual differences, suggesting potential adversarial triggers. The system flags these abnormalities before the image is used in the federated model training, blocking potential backdoor attacks from impacting the global model. The residual value threshold is set to an intensity level of 50 (from the default intensity level of 0) so that normal variations are not mistaken for attacks.

The Neural Cleanse layer synthesizes potential trigger patterns by maximizing the activation values of neurons in the Aggregation layer. Patterns are evaluated on their ability to cause misclassifications when applied to clean inputs, and patterns with high likelihood of being a true backdoor trigger are pruned. These trigger candidates are then stamped on inputs and passed through the trained victim global model to identify neurons that exhibit high activations. The unlearning process in SecFedDrive takes place when a suspicious backdoor pattern is detected by the Neural Cleanse and Residual Check layers. Once the residual check identifies an image with potential adversarial characteristics, the system removes the image from the dataset. Following this, the model undergoes an unlearning process where weights associated with the malicious trigger are adjusted. Backpropagation is leveraged to minimize activations from malicious patterns detected. Specifically, any neuron that becomes hyperactive due to the malicious input is pruned, and the corresponding weights are retrained, ”unlearning” the backdoor while preserving the model’s overall performance. The SecFedDrive architecture maintains the integrity of the federated model by continuously monitoring and correcting these patterns.

Participants	Training Type	FADNet (Loss)		SecFedDrive (Loss)		Attack Success Rate	
		Udacity	CARLA	Udacity	CARLA	FADNet	SecFedDrive
10 Benign, 0 Attackers	Standard	0.107	0.203	0.102	0.193	-	-
9 Benign, 1 Attacker	Backdoor Attack	0.097	0.192	0.103	0.192	93.58%	**1.96%**
7 Benign, 3 Attackers	Backdoor Attack	0.104	0.200	0.101	0.192	98.25%	**2.04%**
5 Benign, 5 Attackers	Backdoor Attack	0.106	0.202	0.102	0.195	99.73%	**2.10%**

TABLE I: Loss performance and attack success rates for FADNet and SecFedDrive on the Udacity and CARLA datasets, with varying numbers of benign and malicious participants. While attack success rates increase with more attackers, SecFedDrive consistently mitigates backdoor attacks across all scenarios.

V. Experimental Results

This section presents an analysis of the performance and robustness of federated learning architectures in autonomous vehicle perception tasks with backdoor attacks. We first outline the experimental setup before evaluating the efficiency of the backdoor attack and respective defense mechanisms.

A. Data Description

The Udacity dataset [14] is a comprehensive collection of 404,916 training and 5,614 testing grayscale video frames captured from real-world urban roads, along with corresponding steering angle labels. Derived from the open-source Udacity self-driving car dataset, it presents a challenging environment with severe lighting changes, sharp curves, and busy traffic scenarios. We take a sub-sample to create our model training dataset, consisting of 39,087 images with a reshaped image size of 32x32 pixels. The CARLA dataset [2] is an augmented dataset of images created using the Carla driving simulator. Different lightings, weather conditions, and driving scenarios were simulated while generating the the 73,235 samples. This dataset also includes a steering angle label for each image.

B. Experimental Setup

The Flower framework [15] used in our experimentation is a helpful decentralized, Python-based approach to deep learning that enables collaborative model training between clients while preserving data confidentiality. It is particularly relevant in the context of FL for autonomous vehicles, as it allows for the training of machine learning models without the need to centralize sensitive data, such as images, sensor readings, or other information collected by individual vehicles. Additionally, we use PyTorch to implement the various layers specified in the architecture as shown in Figure 3.

As seen in Table I, our experiments test on a FL network with 10 participants with 4 test cases for 0, 1, 3, and 5 attackers among them. We also compare the performance of two federated learning architectures, FADNet [2] and our proposed SecFedDrive, on autonomous vehicle perception tasks, using the Udacity and CARLA datasets to evaluate their performance during a data poisoning backdoor attack. To determine when to begin adding poisoned images with pixel backdoors to the model, there is a loss threshold that must be met during the training process. For the Udacity and CARLA datasets, the threshold values are 0.4 and 0.3, respectively.

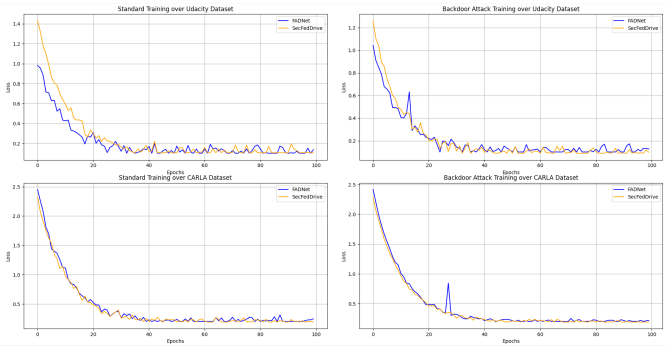

Fig. 4: Loss curves for standard training (left) and backdoor attacks (right) on the Udacity and CARLA Dataset for a given client. The backdoor attack curves notably spike where the model begins to receive infected images when under a predetermined threshold for loss. Otherwise, both models demonstrate comparable curves due to the similarity in architecture implementation between FADNet and SecFedDrive.

C. Attack Efficiency

Results for the various experiments conducted are displayed in Figure 4 and Tables I & II. Table I reports the loss values achieved by each architecture under standard federated training with only benign participants, as well as when subjected to backdoor attacks with varying numbers of attackers. As seen in Fig. 4, both FADNet and SecFedDrive exhibited reasonable loss values around 0.1-0.2 for the benign setting, with comparable loss curves due to their similar architectures. However, when only a single attacker was introduced, the success rate of the backdoor for FADNet reached 93.58%, indicating a significant vulnerability. As the number of attackers increased to 3 and 5, FADNet's attack success rate rose to 98.25% and 99.73% respectively—nearly complete compromise.

Backdoor attacks aim to maintain normal performance on benign data to avoid detection. As the number of attackers performing backdoor attacks increases in the network, FADNet's loss values start to mimic the standard benign training loss. The loss decreases to 0.097-0.192 with one attacker, but then increases again to 0.104-0.202 for three and five attackers. This phenomenon occurs due to the injection of the hidden trigger into the *global* model during federated training; i.e., as more attackers participate, their poisoned updates gain greater influence over the aggregated global model [8]. The global model learns the features of the attacker's trigger pattern as more attacking data is added during training. Thus, the loss of clean inputs converges to values similar to standard training as the model is no longer able to distinguish between the trigger pattern and clean data. Hence, a successful attack is indicated by model predictions overridden with the attacker's label.

D. Defense Efficiency

Table II reports the attack success rates achieved against SecFedDrive when employing different defense strategies. The

Participants	Residual	Neural Cleanse	Combined
9 Benign, 1 Attacker	32.42%	4.88%	1.96%
7 Benign, 3 Attackers	33.65%	5.01%	2.04%
5 Benign, 5 Attackers	33.84%	5.43%	2.10%

TABLE II: Attack success rates (%) for SecFedDrive for different defense mechanisms. The efficacy of Neural Cleanse is enhanced with the residual defense. Using both defense mechanisms, success rates are minimized to approximately 2% for all three participant cases with an adversary.

Residual Check defense, which removes image data that deviates significantly from adjacent frames, reduces the attack success rate over 60% to around 32-34% across varying numbers of attackers. However, this approach still left a substantial vulnerability. Incorporating the Neural Cleanse technique to detect and unlearn backdoor triggers drastically improved SecFedDrive's security, with attack success rates dropping to 4.88-5.43% using Neural Cleanse alone. The most robust defense combined Residual Check and Neural Cleanse, decreasing the attack success rates to only 1.96-2.10%. This multi-pronged defense strategy effectively neutralized the threat of backdoor attacks, securing the integrity of the FL system.

E. Training Time Comparison

Table III compares client and server training times for FADNet and SecFedDrive using Udacity and Carla datasets. For client training, FADNet demonstrates significantly longer training times, with 340 seconds for Udacity and 518 seconds for Carla, whereas SecFedDrive achieves 137 seconds for Udacity and 244 seconds for Carla. Server training is also more efficient for SecFedDrive, reducing time requirements from 636 to 398 seconds (Udacity) and 1097 to 670 seconds (Carla). The reduced server training times with SecFedDrive can lead to significant savings in computational resources and time, facilitating faster deployment of updated models and improving overall system scalability. Additionally, longer training gives attackers more time to introduce poisoned images into the data stream, hence SecFedDrive is a more favorable option.

	FADNet		SecFedDrive	
	Udacity	Carla	Udacity	Carla
Client Training	340 sec.	518 sec.	137 sec.	244 sec.
Server Training	636 sec.	1097 sec.	398 sec.	670 sec.

TABLE III: Comparison of client and server training across different architectures and datasets, with SecFedDrive demonstrating substantial improvements over the FADNet architecture.

VI. Conclusion

This paper explored the vulnerabilities of FL systems to backdoor attacks and tested countermeasures in the context of autonomous driving. Extensive experimentation demonstrated the effectiveness of different backdoor strategies and highlighted the significant threat they pose to FL models. It is seen that as the number of attackers in the network increases, the loss values for the vulnerable FADNet architecture mimic those of standard benign training, indicating more success and stealth in the attackers' manipulation of the model. To counter these threats, we proposed and evaluated the use of Residual Check and Neural Cleanse as defense layers in the learning architecture, with combined use being the most effective. In future works, we aim to develop additional defense mechanisms like Februus [16] and test the architecture's durability against new attacks such as [17].

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[2] 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 *2022 IEEE Intelligent Vehicles Symposium*, 2022, pp. 1824–1830.

[3] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[4] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning*. PMLR, 2019, pp. 634–643.

[5] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2019.

[6] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International conference on artificial intelligence and statistics*. PMLR, 2020, pp. 2938–2948.

[7] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 707–723.

[8] J. Tu, T. Wang, J. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "Adversarial attacks on multi-agent communication," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[9] E. Bagdasaryan and V. Shmatikov, "Blind backdoors in deep learning models," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 1505–1521.

[10] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "Crfl: Certifiably robust federated learning against backdoor attacks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 372–11 382.

[11] C. Wu, X. Yang, S. Zhu, and P. Mitra, "Mitigating backdoor attacks in federated learning," *arXiv preprint arXiv:2011.01767*, 2020.

[12] S. I. Nanayakkara, S. R. Pokhrel, and G. Li, "Understanding global aggregation and optimization of federated learning," *Future Generation Computer Systems*, 2024.

[13] H. Li, W. Luo, X. Qiu, and J. Huang, "Identification of various image operations using residual-based features," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 1, pp. 31–45, 2016.

[14] Z. Yang, Y. Zhang, J. Freyberger, N. Roberts, J. Kan, and S. Athey, "Learning to steer by mimicking features from heterogeneous auxiliary networks," in *Proceedings of the 5th Annual Conference on Robot Learning*, 2022.

[15] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[16] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Proceedings of the 36th Annual Computer Security Applications Conference*, 2020.

[17] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 966–11 976.