

Federated Learning Under Attack: Exposing Vulnerabilities through Data Poisoning Attacks in Computer Networks

Ehsan Nowroozi , *Senior Member, IEEE*, Imran Haider , *Member, IEEE*, Rahim Taheri , *Member, IEEE*,
Mauro Conti , *Fellow Member, IEEE*

Abstract—Federated Learning (FL) is a machine learning (ML) approach that enables multiple decentralized devices or edge servers to collaboratively train a shared model without exchanging raw data. During the training and sharing of model updates between clients and servers, data and models are susceptible to different data-poisoning attacks.

In this study, our motivation is to explore the severity of data poisoning attacks in the computer network domain because they are easy to implement but difficult to detect. We considered two types of data-poisoning attacks, label flipping (LF) and feature poisoning (FP), and applied them with a novel approach. In LF, we randomly flipped the labels of benign data and trained the model on the manipulated data. For FP, we randomly manipulated the highly contributing features determined using the Random Forest algorithm. The datasets used in this experiment were CIC [1] and UNSW [2] related to computer networks. We generated adversarial samples using the two attacks mentioned above, which were applied to a small percentage of datasets. Subsequently, we trained and tested the accuracy of the model on adversarial datasets. We recorded the results for both benign and manipulated datasets and observed significant differences between the accuracy of the models on different datasets. From the experimental results, it is evident that the LF attack failed, whereas the FP attack showed effective results, which proved its significance in fooling a server. With a 1% LF attack on the CIC, the accuracy was approximately 0.0428 and the ASR was 0.9564; hence, the attack is easily detectable, while with a 1% FP attack, the accuracy and ASR were both approximately 0.9600, hence, FP attacks are difficult to detect. We repeated the experiment with different poisoning percentages.

Index Terms—Federated learning, Causative attacks, Adversarial machine learning, Corrupted training sets, Cybersecurity, Data-poisoning

I. INTRODUCTION

IN this modern era of technology, in which other scientific disciplines are advancing swiftly, FL is also keeping pace

and progressing rapidly. In the FL setting, a central server collects information from different connected clients regarding their data in a secure manner, such as gradient updates. In this setup, instead of sharing data with a central server, only an updated model is sent or received by a client in the form of gradients or model parameters [3]. Sending data over a network causes data privacy issues and communication overheads. To overcome these challenges, FL techniques allow gradient information to be sent instead of raw data [4]. However, sending model updates over a network can also unknowingly leak sensitive information to a central server or any other third party [5]. FL has many practical applications and is used in various distributed systems, such as Google's board [6], to predict the next word while typing. In addition, FL potential applications include sentiment assessment, context-driven points, responding to pedestrian actions in self-driving cars, and foreseeing health incidents such as heart attack susceptibility through wearable gadgets [7]. The wide adaptability of FL across multiple fields has significantly amplified its allure from the perspective of potential threats. While conducting attacks against FL, different threat models are considered, including the objective of the adversary, knowledge of the adversary, and capabilities of the adversary [8]. The objective of an adversary is to violate the integrity and availability of a model. The adversary can have complete knowledge of the global model, also known as the white-box setting, in which the adversary can access the data of all collaborating clients, the global model parameters, and its predictions. However, in a black-box setting, the attacker does not have any information regarding the data or model parameters [9]. In another case, the adversary can have partial knowledge of the FL setting, which means that the attacker can have access only to the local data of compromised clients but not of benign clients [3].

In general, attacks in FL can be divided into two categories (i) causative attacks [10], and (ii) exploratory attacks [11]. Causative attacks affect the learning process by controlling training data. These attacks include the injection of training data with erroneous labels, which can reduce the accuracy of the trained model. Typically, these attacks occur prior to model training [12]. However, exploratory attacks capitalize on misclassifications but do not influence the training procedure [13]. There are different kinds of causative attacks that can be carried out against FL such as data poisoning attacks [14], model poisoning attacks [15], membership inference attacks [16], and many more. These attacks can be classified into

E. Nowroozi is with the Centre for Secure Information Technologies (CSIT), Queen's University Belfast, Northern Ireland, United Kingdom (e-mail: ehsan.nowroozi65@gmail.com, e.nowroozi@qub.ac.uk) - Corresponding Author is Ehsan Nowroozi.

I. Haider is with the Department of Natural Engineering and Sciences, Bahcesehir University (BAU), Istanbul, Turkey (e-mail: imran.haider@bahcesehir.edu.tr)

R. Taheri is with the School of Computing, Faculty of Technology, University of Portsmouth, United Kingdom (e-mail: rahim.taheri@port.ac.uk)

M. Conti is with the Department of Mathematics, Security and Privacy Research Group, University of Padua, 35121 Padua, Italy, and also with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: mauro.conti@unipd.it).

various subcategories. There are two types of membership inference attacks: (i) active and (ii) passive. In an active membership inference attack, the attacker could be one of the participants in federated learning who adversarially modifies his parameter uploads W_t^i or could be the central aggregator who adversarially modifies the aggregate parameters W_t that he sends to the target participant(s). However, in a passive membership inference attack, an attacker can only observe genuine computations using the training algorithm and model [17]. Attackers can use both approaches to compromise the clients and servers. [16].

In this study, our main focus was on data poisoning attacks. We implemented two instances of data-poisoning attacks: LF and FP. In LF attacks, the labels of some proportion of the training dataset are flipped so that the attack remains undetectable by humans; for example, if it is 0, then it is changed to 1, and vice versa. By contrast, in FP, we can manipulate some values of the most important columns of the data, and we can use different techniques for it, such as replacing them with the mean or mode of that column.

Most importantly, we determined that the LF attack is not effective in FL, specifically in Computer Networks. The reason for this is the drastic drop in the server accuracy from 90% to approximately 10% after applying the LF attack, making it easy for the system to understand that there is an attack. Moreover, the ASR also increased to almost 90%, which makes the attack more suspicious, and we cannot fool the system by applying this attack to the client because the system can easily detect it. On the other hand, the FP attack was successful in fooling the system because there was no major downward transition in the accuracy of the server, whereas the ASR exhibited a radical and significant upward shift, and this attack remained undetectable to the system, making it highly dangerous.

A. Contributions

The following points highlight the contributions of this study:

- In this study, we conducted training sessions for neural networks using two prominent datasets in the field of computer networks, namely the CIC and UNSW datasets. Our choice of these datasets was driven by their widespread use and relevance in the domain. The primary objective of our neural network training was to perform binary classification of network traffic, distinguishing between benign (0) and malicious (1) traffic. Through these efforts, we aimed to contribute valuable insights and advancements to the understanding and detection of malicious activities in computer networks.
- In the FL configuration employed for this study, our experimental setup comprised two clients and a server. Specifically, in a white-box scenario, we designed and implemented two data-poisoning attacks, namely label flipping (LF) and feature poisoning (FP), exclusively on one of the clients. The objective was to assess the repercussions of these attacks on the accuracy of the server. Our investigation into these targeted adversarial

techniques contributes valuable insights into the vulnerabilities and potential mitigations within the FL framework.

- Our study delves into the efficacy of data poisoning attacks within the computer network domain, adopting an innovative approach by implementing these attacks at varying percentages on the training data of a single client. This experimentation, conducted on datasets specifically curated for the computer network domain, enabled us to assess the impact of such attacks comprehensively. We calculated both server accuracy and Attack Success Rate (ASR) across different attack percentages, providing a nuanced understanding of the vulnerabilities introduced by data poisoning. To facilitate the reproducibility and adaptability of our experiments, we developed a highly flexible and generic codebase. This codebase allows for seamless adjustment to diverse datasets and facilitates experimentation with different attack percentages. Our contribution lies not only in the insights gleaned from our experiments but also in the provision of a tool that can be easily employed and extended for future research in this area.
- In our investigation, we recorded the server accuracy and Attack Success Rate (ASR) across varied poisoning scenarios for both datasets. Our findings underscore the effectiveness of data poisoning attacks when implemented at different percentages, revealing their impact on the accuracy of the server. Additionally, we provide a comprehensive analysis of the incurred loss experienced by both clients, shedding light on the broader consequences of these attacks. This contribution elucidates the nuanced dynamics of data poisoning in our experimental setting, offering valuable insights into the resilience and vulnerabilities of the system under examination.

B. Organization

We outline the rest of our paper as follows: In Section II, we provide an overview of related works that discuss data-poisoning attacks using different datasets. In Section III, we discuss the datasets and the network architecture of our BAU1 model. This section also includes the experimental setup and techniques that we applied to perform both data-poisoning attacks. Section IV presents the results of our experiments performed under different scenarios. In Section V we summarize our study and propose future work in the field of FL.

II. RELATED WORKS

In data poisoning, different types of attacks are used by researchers and adversaries to manipulate data, such as two important data-poisoning attacks, LF and FP. LF attacks have extensive applications in image processing, as in [18] authors evaluated the performance of LF attacks and proposed a defense mechanism on two real datasets from the UCI repository [19]: MNIST [20] and Spambase [21], which are common benchmarks for classification tasks. Furthermore, in [22] the authors used an LF attack to manipulate the MNIST dataset

and trained a CNN on a benign and label-poisoned dataset. They found that the attack slightly increased the classification error after injecting ten poisoning points into the training data. The experiment was repeated using the same setting. They used a Multiclass Logistic Regression classifier instead of a CNN and found that the error increased from 2% to 2.1% after a random LF attack. The approach outlined in [23] can be readily expanded to encompass a label-specific scenario, where the adversary can adjust the predictions of the victim classifiers based on predetermined rules as opposed to merely generating incorrect predictions. Experiments were repeated on various datasets, including CIFAR-10 [24] and a reduced version of ImageNet, and the effectiveness of the proposed method was confirmed.

Poisoning attacks are extensively employed for model poisoning and data poisoning purposes, as in [25] they used different flavors of state-of-the-art data poisoning attacks such as random LF, random label and FP, label swapping, and FP attacks. Moreover, LF attacks are widely used by researchers owing to their simple nature. This attack was prominent in this study [26] in which they carried out it on the CIFAR-10 and Fashion-MNIST datasets. They introduced a small percentage of malicious participants, ranging from 2% to 50%, that contained malicious training data manipulated using an LF attack. For example, in CIFAR-10 image classification, airplane \rightarrow bird denotes that the airplane image label is changed to a bird. This attack was intended to boost the likelihood of the global model making incorrect classifications, especially by leading to more frequent misclassifications of airplane images as bird images during testing, and they achieved effective results.

Label poisoning was introduced in this study [27] and was generated through GANs and used to train local models. The datasets used were UNSW [2] and NIMS [28] which are related to computer networks. Furthermore, they proposed two defense methods to mitigate LF attacks on the FL. In addition to the LF attack, another data-poisoning attack known as the FP attack plays a critical role in FL security. Many datasets have a massive number of features, ranging from hundreds to thousands, and selecting the features to poison is a critical challenge known as feature importance. ML feature selection is also widely used for generalization and has both advantages and disadvantages.

We have advanced subcategories of data-poisoning attacks that can also be used by attackers in FL, such as generative adversarial network (GAN) attacks [29]. In this type of attack, the adversary attempts to generate data that closely resembles legitimate data in a collaborative learning environment. However, there are two limitations to this type of attack: the attacker should have background information regarding the victim's data, and the other requirement is that all members of the class are similar [30]. Furthermore, another advanced type of data poisoning attack aims to produce perturbed samples in the training data using auto-encoders (AE). An auto-encoder (AE) is a type of neural network that tends to recreate its input data, and adversaries can use it to generate malicious samples for training data [23].

In their research [31] they demonstrated that Intrusion Detection Systems (IDS) for IoT based on FL are prone

to backdoor attacks. In their proposed attack strategy, the threat actor(s) can fool the detection model by employing compromised IoT devices to introduce minimal quantities of poisoned data during the training procedure while remaining undetected throughout the process. Moreover, researchers are actively working on new defense methodologies to reduce data-poisoning attacks. However, more work is required to make FL systems more robust to these attacks, which also evolve. In Table I we have compared the work of other researchers with ours and listed the model(s) used, datasets utilized, strengths, and weaknesses.

III. METHODOLOGY

In this study, we used the two most popular datasets, CIC [1] and UNSW [2] related to computer networks. We trained two DL models on these two datasets separately and captured their accuracy. In this FL setup, we considered just two clients and a server and then also applied two data poisoning attacks, LF and FP to only one client. In DL, an attacker can modify the training dataset based on knowledge and information. According to the currently existing literature, there exist three scenarios that an adversary can consider to conduct adversarial attacks: white-box, gray-box, and black-box attack scenarios [17]. The adversary possesses perfect knowledge (PK) about the training data and model in a *white-box* setting and can generate adversarial instances for training and modify the model updates. In the case of a *gray-box* scenario, the threat actor has limited knowledge (LK) of the training data and model. The adversary does not know the internal information of the system in a *black-box* setup, which is a more viable and complex case than the other scenarios. Consequently, the attacker employs recurring inquiries to gather such sensitive data. We carried out this experiment in a *white-box* scenario as we had access to both data and model. Additionally, a very simple illustration of our methodology is given in Figure.1. Detailed information about the datasets and data poisoning

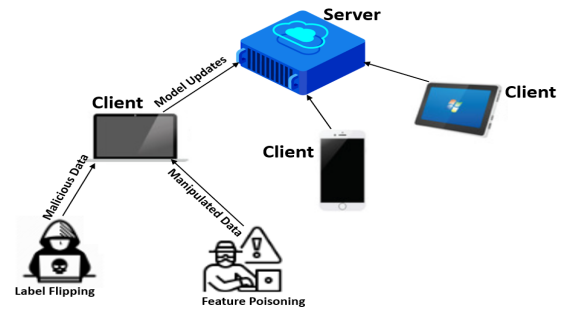


Fig. 1. Illustration of Adversarial Data Manipulation: A Server and Two Clients. Trained CL1 on Manipulated Data with LF and FP Attacks

attacks is provided below:

A. Datasets

To examine the effect of adversary capability on models by data poisoning in computer networks, we used the most popular datasets, the CIC [1] and the UNSW [2] that are part of many research works [27], [37], [38]. We used them to train

TABLE I
COMPARISON BETWEEN RELATED AND OUR WORK

Ref.	Model(s)	Datasets	Strength(s)	Weakness(es)
[18]	K-NN	MNIST [20] Breast Cancer [32], SpamBase [21]	- Proposed Defense mechanism against LF attack	- Used a very simple linear classifier - Defense mechanism is not effective for other types of poisoning attacks
[22]	LR MLP	SpamBase [21] Ransomware [33]	- extension of poisoning attacks from binary to multiclass problems. - attack transferability is also possible.	- High complexity of Back-Gradient Optimization
[23]	AE	CIFAR10 [24] ImageNet [34]	- Transferability Across Classifiers	- Dependence on Hyperparameters
[25]	CNNs	MIT-BIT arrhythmia [35] HAR [36]	- Provide framework to detect poisoning attacks - low computations complexity	- White-box assumption - Data privacy concerns.
[27]	GANs	UNSW [2] NIMS [28]	- Proposed defense against LF attack - combined generative adversarial schemes with noisy-label classifiers	- Scalability and efficiency concerns - Adversarial training, GANs, and noisy-label classifiers computationally complex.
[29]	GANs	MNIST [20] CIFAR-10 [24]	- Legitimate data generation - Generative Method for Accelerated Attack Generation	- Limited Validation on Diverse Datasets - Background information on the victim's data is required.
Ours	NNs	UNSW [2] CIC [1]	- LF and FP attack on two computer networks datasets	- Effective in white-box scenario - Lack of discussion on defense mechanisms

the model on both benign and malicious examples. Therefore, we used two datasets that are highly related to our research domain. We gathered these datasets in the form of pcap files and then extracted those pcap files using NFStream [39]. NFStream employs nDPI-based deep packet inspection to identify encrypted applications and perform metadata fingerprinting accurately. This includes the detection of protocols such as TLS, SSH, DHCP, and HTTP. nDPI stands for "nTop Deep Packet Inspection." It is an open-source library and toolkit used for deep packet network traffic inspection. Deep packet inspection involves analyzing the content of network packets at a granular level to understand the protocols, applications, and services being used on a network. nDPI is commonly used in network monitoring and security applications to classify and analyze network traffic, making it easier to detect and respond to security threats, optimize network performance, and gain insights into network usage patterns.

We trained the clients on both CIC and UNSW and in all scenarios, divided the dataset in this way: 80% of the data for the training and 20% of data for testing and validation. Moreover, each dataset contains around one million data samples. Furthermore, partitioned the training dataset into two segments, allocating one segment to CL1 and the other to CL2. Additionally, subdivided 20% of testing data into two separate sets, distributing 10% for validation and reserving the remaining 10% for testing. Before giving data to the model, also pre-processed it for efficient results such as filling the missing values with a specified number. If the number is not specified then those values are filled with 0. In addition, during the pre-processing of the data normalization was applied because it gives improved convergence, enhanced model performance, equalized influence, and interpretability because normalized data has consistent scales, that simplify the comparison of feature importance.

B. Network Architecture

We adopted a single deep-learning model for both clients during the training and testing phases. Initially, this same model was utilized for our server as well. This model was composed of an input layer, and the number of neurons in this layer is set to the feature size, where feature size represents the number of (Total_Columns-1) in the dataset, and we subtracted one column because one column represents the output label column and is not part of the feature columns. Its number of output neurons was 2048. The model also contains two hidden layers, one of which contains 2048 input neurons, whereas the number of output neurons is 1024, which is the number of input neurons for the next hidden layer. Similarly, there are two output neurons in the second hidden layer, which is equal to the number of output classes. Moreover, the model was also constructed from two activation functions and used the Rectified Linear Unit (ReLU) activation function named ReLU1, which was applied after the first hidden layer, and ReLU2, which was applied after the second hidden layer. The architecture of the neural network model, named BAU1, can be easily obtained from Figure 2. Furthermore, batch

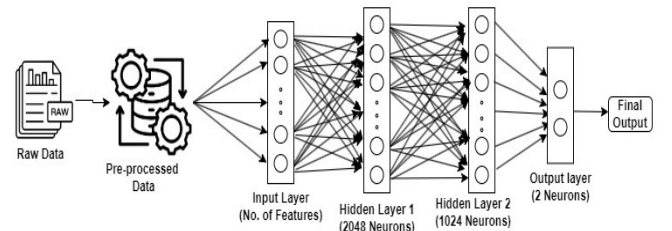


Fig. 2. Architecture of Neural Network Model (We referred this network to as BAU1)

normalization was applied to both hidden layers. Normal-

ization is performed during pre-processing and can lead to more stable and effective ML models. A dropout layer with a specific dropout probability (self.dropout_p) is also used as part of the neural network architecture. During training, it randomly deactivates some neurons to increase the network's proficiency for generalization and diminish overfitting. The output activation function that we used was the log softmax function, which computes the natural logarithm (base e) of softmax probabilities. The cross-entropy loss function is used in this network, which is commonly used for classification problems to compute the loss between the predicted probability distribution and the labels of the true class. The output layer of our model comprises two neurons, and each neuron represents one of the binary classes (0 or 1), where 0 represents the benign class and 1 corresponds to the malicious class.

C. Experimental Setup

To build the model, we considered 8,38,861 samples for training, 1,04,857 for validation, and 1,04,857 for the test set. In Table II different attack scenarios are presented, in which $N_{BAU1-LF}^{UNSW}$ represents the NN model named BAU1, which is trained with the UNSW dataset, and to this dataset, the LF attack was applied. Similarly, this scenario, $N_{BAU1-FP}^{CIC}$, indicates that we trained BAU1 with the CIC dataset that is poisoned using the FP attack. In addition, another important scenario $N_{BAU1-LF}^{CIC}$ points to the LF attack on the CIC dataset, which is used to train the BAU1 model, whereas $N_{BAU1-FP}^{UNSW}$ denotes the FP attack on the UNSW dataset that is fed to the BAU1 model for training. The number of samples in both datasets is also equal, that is, 10,48,575 examples in each dataset. Most DL models are designed to accept images as input, which are usually three-dimensional, and the dataset we used was one-dimensional. Therefore, to make the model feasible, two additional dimensions were added. After adding extra dimensions, the shape of the np array was changed, and it was performed for both clients' training data and testing and validation data. We developed an NN using PyTorch [40] which is a popular Python library. We performed our analysis using hardware specification MSI GF65. We have made all our implementation codes publicly available in the GitHub repository [41]. For the NN model, we considered 20 epochs for training using the Stochastic Gradient Descent (SGD) optimizer with a random learning rate between 1×10^{-4} and 9.9×10^{-3} and a momentum of 0.9. The batch size for training and validation is set to 1000. For both clients, we used the Federated Averaging (FedAvg) algorithm [42] which is a common setup in FL, where multiple clients with local datasets train models in a collaborative manner while preserving data privacy.

D. Empirical Study

In this experiment, we performed two data-poisoning attacks on two models that were trained on two different datasets. First, an LF attack was applied. Before performing this attack, we trained the model with a benign dataset and captured the results, saved the models, and tested the accuracy of the saved models using test data. We then poisoned the data by

flipping the labels for 1% of the data. Again, the model was trained with malicious data, and the results and models were saved. Here, to compute the ASR, we flipped the labels of the complete test data and noted the accuracy that indicates the strength of the attack. We repeated the experiment with 2%, 3%, 4%, 5%, 7%, 10%, 15%, 20%, and 25% poisoned data, documented the results for all these attack percentages, and computed the ASR for all of them.

We performed the same experiment for both the CIC and UNSW datasets. Our code is written in such a generic manner that simply changes the name of the dataset; the rest of the processes, for example, pre-processing of data, splitting the training data for both clients, training of the model with/without attack, computation of ASR, and saving the results in an Excel file, are performed for both datasets without any user interaction. For the FP attack, we introduced some new changes in the code; however, we used a random forest algorithm [43] to compute feature importance. In the CIC dataset, the *first column* has the highest feature importance, which we computed using permutation on the full model, as shown in Figure. 3. Permutation-based feature importance is an approach that involves logically rearranging the values of each feature, analyzing the influence on the performance of the model, and hence determining the impact intensity of each feature in the decision-making step. We used a similar strategy

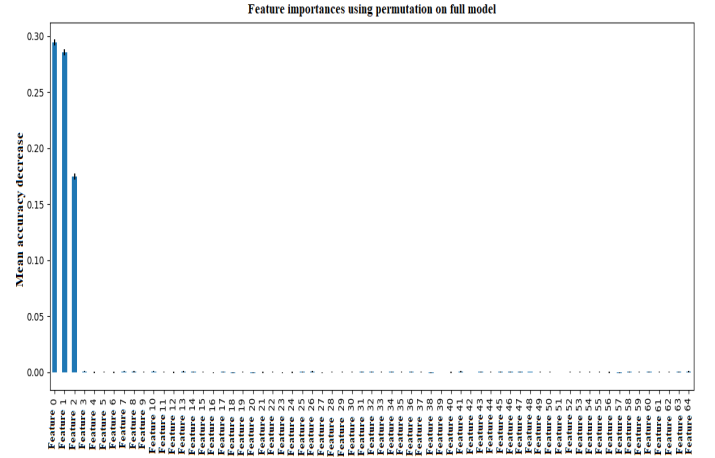


Fig. 3. CIC: Feature importance using Permutation on the full model

to determine the feature importance in the UNSW dataset and found that *second column* has the highest feature importance. The graph of all columns with their feature importance is given below for this particular dataset in Figure 4. In FP, we manipulated the values of the most important features that were determined using the random forest technique. There were two label classes in both datasets: 0 and 1. We first calculated the *mean* of all values where the label is 0 and computed the mean of values with label 1. In the next step, we found the minimum and maximum values in the feature column that are different for the two datasets. Subsequently, the values of the most important features were normalized using the *min-max* normalization technique. We changed the values of the column where the label was 0 to random unique values, where the label was 1. Detailed information

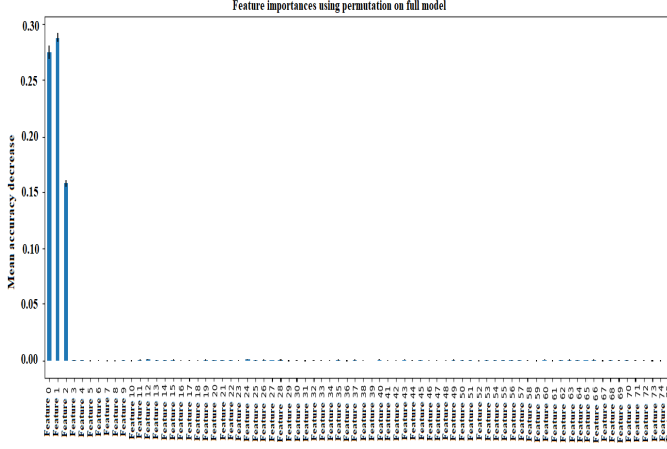


Fig. 4. UNSW: Feature importance using Permutation on the full model

regarding our FP attack can be found in Algorithm 1. In this algorithm, D represents a dataset that contains feature column values. att_per represents the attack percentage which is also represented as P in the algorithm, which is the percentage of values that are to be manipulated. min_value and max_value represent the minimum and maximum values in the feature column, respectively. The i represents the iterator value of the loop, L represents the value of the label that may be either 0 or 1, and $percent$ represents the number of values to be manipulated in the feature column. Additionally, $number$ represents the user-defined att_per , and $column_index$ is the index of the target feature column.

TABLE II
ATTACK SCENARIOS ON BAU1 NETWORK WITH DIFFERENT DATASETS

Scenarios	Dataset	Attack Strategy
$N_{BAU1-LF}^{UNSW}$	UNSW	Label Flipping
$N_{BAU1-FP}^{UNSW}$	UNSW	Feature Poisoning
$N_{BAU1-LF}^{CIC}$	CIC	Label Flipping
$N_{BAU1-FP}^{CIC}$	CIC	Feature Poisoning

In contrast, to calculate the ASR, the value of the column where the label was 0 was replaced with the normalized average value of label 1 and vice versa. Because we considered two clients in this experimental setup, we performed both attacks on only one client, which we named CL1. The percentage of attacks was calculated as the number of values to be manipulated according to the desired percentage value. We set up an array containing integer values. In every iteration, if the iterator value is not in that array, then the iterator value is considered a percentage number. This percentage is multiplied by the number of labels in the training data of CL1, which is also equal to the number of samples/rows in the training data. In equation form, it can be written as:

$$number_of_values = \left\lfloor \frac{\text{len}(CL1_Y) \cdot \text{attack_percentage}}{100} \right\rfloor \quad (1)$$

In the above equation, $CL1_Y$ represents the length of the array of labels of training data for CL1, where len in Python

Algorithm 1: Feature Poisoning (FP) Attack

Data: D : Feature column values,
 L : Labels (0 and 1),
 att_per : P
Result: Transformed feature values

- 1 **Step 1:** Find the min and max values in feature column;
- 2 $min_value \leftarrow \min(D)$;
- 3 $max_value \leftarrow \max(D)$;
- 4 **Step 2:** find the average for label classes 0 and 1;
- 5 $average_zero \leftarrow \text{average}(D \text{ where } L = 0)$;
- 6 $average_one \leftarrow \text{average}(D \text{ where } L = 1)$;
- 7 **Step 3:** Normalize the values;
- 8 **for** i **in** dataset **do**
- 9 **if** $L = 0$ **then**
- 10 $normalized_value \leftarrow \frac{average_zero - min_value}{max_value - min_value}$;
- 11 **else**
- 12 $normalized_value \leftarrow \frac{average_one - min_value}{max_value - min_value}$;
- 13 Update the feature value with $normalized_value$;
- 14 **Step 4:** Modify malicious samples to benign samples;
- 15 $number \leftarrow P$;
- 16 $percent \leftarrow \text{int}(\text{len}(L) \times (number/100))$;
- 17 **for** $i = 1$ **to** $percent$ **do**
- 18 $random_index \leftarrow \text{random}(0, \text{len}(\text{unique_values}) - 1)$;
- 19 **if** $L[i] = 1$ **then**
- 20 $D[i, 0, 0, column_index] \leftarrow \text{unique_values}[random_index]$;

is used to determine the length of an array, while $_of_attack$ represents the percentage number, and the $number_of_values$ denotes the number of values that we have to change in the feature column. During computations in Python, this equation can return the float value, so that it changes to the integer value we have used the floor function.

IV. RESULTS AND DISCUSSION

In this section, we discuss the results obtained during the experiment using different scenarios, with and without an attack.

A. FL model in the presence of NO attack

We trained the BAU1 model using benign datasets. First, we trained on the CIC dataset without an attack, and reported the results, which are represented as N_{BAU1}^{CIC} . We used this notation in Table III to clarify the results. Subsequently, we trained the BAU1 model with the UNSW dataset without an attack, which is denoted as N_{BAU1}^{UNSW} in Table III. We captured the server accuracy for both the scenarios. Moreover, we had

two clients during this experimental study and recorded their losses, as listed in Table III. In this table, we report the loss of

TABLE III
RESULTS WITHOUT ATTACK

Scenario	Component	Loss	Accuracy
N_{BAU1}^{CIC}	Client-1 & Client-2	0.6406	-
N_{BAU1}^{CIC}	Server	-	0.9680
N_{BAU1}^{UNSW}	Client-1 & Client-2	0.6787	-
N_{BAU1}^{UNSW}	Server	-	0.8027

client-1 and client-2 and save the values obtained after the last epoch. The loss value for both clients is 0.6406 for the CIC dataset. In the same scenario, the server accuracy was 96.8

B. Results With Label Flipping (LF) Attack on CIC dataset

Table IV lists the results of the LF attack on the CIC dataset. Before, explaining the results of this scenario, there are some common things in all the result tables in all attack scenarios. We set a threshold value for the successful results, which is 0.40, which means that if the server accuracy and ASR are equal to or greater than 0.40, then the attack percentage will be considered a successful case.

In Table IV it is clear that the server accuracy drops significantly and the ASR increases; therefore, an LF attack creates a situation of suspiciousness in the system. The losses for client-1 and client-2 were 0.7352 and 0.7361, respectively. With 1% data manipulation, the accuracy of the server was transcribed as 0.0428, whereas the ASR is 0.9564, indicating that the attack failed to fool the server because the accuracy of the server dropped drastically to a very small value and the system could understand it very easily. A similar behavior can be observed with a 2% attack with an accuracy of 0.0537, whereas the ASR is 0.9457, which is also a failure scenario for the LF attack because the attack can no longer act as a ghost in the system and can be detected easily by the system. To further investigate the impact of the LF attack, we increased the attack percentage. We observed new behavior in server accuracy and ASR with 3 % and 4 % attacks. The accuracy of the server with 3% attacks increased to 0.9680, whereas the ASR value dropped to 0.0329, and almost similar results were obtained with a 4% attack. The increase in accuracy is very high, while the ASR value is very low because ASR is not satisfactory in both 3% and 4% of attack cases, and because of the very small ASR values, both cases are considered to fail.

An attack strength of 5% yields interesting results. The accuracy dropped to 0.7739, whereas the ASR increased to 0.2292, which is not a successful scenario according to the threshold. However, detecting the LF attack with this percentage is difficult because the accuracy has not dropped too low, and the ASR showed some improvement. To simplify our experiment, we do not calculate the results for 6%, 8%, or 9%. With 7%, 10%, 15%, and 20% of attacks, we observed similar results, where the accuracy of the server was very low, while ASR had a significantly high value, indicating that these

failed percentages of the attack. With 25%, we again observed the opposite behavior, in which the accuracy of the server increased to 0.9204, while ASR decreased to 0.0797; hence, the attack failed with this percentage of attack because of a very small value of ASR.

TABLE IV
LF POISON ATTACK FOR THE SCENARIO N_{BAU1}^{CIC-LF} ON CIC DATASET

Poison	Component	Loss	Acc	ASR
1%	Client-1 & Client-2	0.7352 & 0.7361	-	-
1%	Server	-	0.0428	0.9564
2%	Client-1	0.7224 & 0.723	-	-
2%	Server	-	0.0537	0.9457
3%	Client-1 & Client-2	0.6378 & 0.7116	-	-
3%	Server	-	0.968	0.0329
4%	Client-1 & Client-2	0.6721 & 0.6713	-	-
4%	Server	-	0.9486	0.0539
5%	Client-1 & Client-2	0.6803 & 0.6794	-	-
5%	Server	-	0.7739	0.2292
7%	Client-1 & Client-2	0.708 & 0.7093	-	-
7%	Server	-	0.1256	0.8720
10%	Client-1 & Client-2	0.7464 & 0.7534	-	-
10%	Server	-	0.032	0.9670
15%	Client-1 & Client-2	0.7129 & 0.7171	-	-
15%	Server	-	0.0447	0.9543
20%	Client-1 & Client-2	0.7076 & 0.7116	-	-
20%	Server	-	0.1281	0.8718
25%	Client-1 & Client-2	0.6744 & 0.6669	-	-
25%	Server	-	0.9204	0.0797

C. Results With Label Flipping (LF) Attack on UNSW dataset

This section discusses various cases of LF attacks on the UNSW dataset. We repeated the experiment with one change in data. Previously, we carried out an LF attack on the CIC dataset; however, we changed the dataset to an UNSW, which is also related to computer networks.

In Table V, starting with a 1% LF attack, the losses of client-1 and client-2 were 0.6791 and 0.6788, respectively. Furthermore, the accuracy of the server with a 1% attack rate was 0.8554, whereas the ASR was only 0.1423. This value of ASR is less than the threshold, which is why we did not consider this percentage of the attack as a successful attack. Although the accuracy was high, the ASR was low. We observed more interesting results with the 2% and 3% attacks. With 2% and 3% LF attacks, the loss of both clients increased to approximately 0.7300, which resulted in a decrease in the server accuracy to approximately 0.1000 and an increase in the ASR to approximately 0.9000. With a sudden drop in accuracy and an immense increase in the ASR, the system will clearly understand that it is under attack, making this case a failed attack. We want to consider only the attack case as a successful attack that gives high values for both the accuracy of the server and the ASR, because we want our attack to act as a ghost in the system. The attack with 4% failed because the ASR value was 0.1917, whereas at 5%, the ASR value increased to 0.7193 and the server accuracy dropped to 0.2815, which also makes it an unsuccessful attack scenario. Attacks of 7%,

10%, and 15% failed because their ASR values were less than 0.1757 in both 7% and 10% and could not cross the threshold value, although the accuracy was greater than 0.8000. In either case, the ASR value is not the desired value. Moreover, the attack was unsuccessful with a 20% LF attack because the ASR value was 0.8212; however, the server accuracy dropped to 0.1795, making it suspicious in the system. However, with 25% attack failure, although the server accuracy was logged at 0.7097, the ASR value was 0.2919, resulting in a failed LF attack percentage. Based on these results, we can conclude that the LF attack does not generate effective outputs and fails to fool the server.

TABLE V
LF POISON ATTACK FOR THE SCENARIO $N_{BAU1}^{UNSW-LF}$ ON UNSW DATASET

Poison	Component	Loss	Acc	ASR
1%	Client-1 & Client-2	0.6791 & 0.6788	-	-
1%	Server	-	0.8554	0.1423
2%	Client-1 & Client-2	0.731 & 0.7317	-	-
2%	Server	-	0.0951	0.9052
3%	Client-1 & Client-2	0.7271 & 0.7258	-	-
3%	Server	-	0.1000	0.8997
4%	Client-1 & Client-2	0.6734 & 0.6725	-	-
4%	Server	-	0.8072	0.1918
5%	Client-1 & Client-2	0.7041 & 0.7047	-	-
5%	Server	-	0.2815	0.7193
7%	Client-1 & Client-2	0.6784 & 0.6771	-	-
7%	Server	-	0.8253	0.1757
10%	Client-1 & Client-2	0.6708 & 0.6676	-	-
10%	Server	-	0.8816	0.1181
15%	Client-1 & Client-2	0.6853 & 0.6835	-	-
15%	Server	-	0.6793	0.3186
20%	Client-1 & Client-2	0.7032 & 0.7059	-	-
20%	Server	-	0.1795	0.8212
25%	Client-1 & Client-2	0.6856 & 0.6826	-	-
25%	Server	-	0.7097	0.2920

D. Results With Feature Poisoning (FP) Attack on CIC dataset

In Table VI there are successful cases that show the effectiveness of the FP attack performed with the CIC dataset on one of the two clients in the FL to fool the server. In this scenario, we considered the poison percentages of the attack as successful, in which both the server accuracy and ASR had values greater than the threshold value, which in this case was set to 0.40. We manipulated the values of one feature of the CIC dataset and trained the model on 1% manipulated values of the most important feature columns in the dataset. With 1% poisoned data, we observed that the accuracy of the server was 0.9642 and, more interestingly, the ASR value was 0.9628. The accuracy of the server was also very high along with the ASR value, which means that in this case, our attack worked perfectly because it is unlikely to be detected by the server, and our attack can act as a ghost in the system. As we will discuss only the successful cases in this paragraph, the next case that was able to cross the threshold values was the

poisoning of 4% of the feature column values. After training, we tested the accuracy of the server, which was determined as 0.8611. In addition, the ASR for this 4% poisoning case was also transcribed at 0.8616, which is a clear indication of a successful attack scenario, because both values were higher than the threshold. Subsequently, a successful case was observed with 20% values of poisoned data in the feature column, where the accuracy of the server was 0.7427, and the ASR value was reported as 0.7763. The next successful case was with 25% of poisoned values in the feature column of the dataset, and after training when the server was tested, the accuracy reached 0.9680, while the ASR also increased to 0.9671.

We discuss some unsuccessful cases in N_{BAU1}^{CIC-FP} scenario, where the first case is transcribed with 2% and 3% of the poisoned data. With 2% feature-poisoned data, the accuracy of the server dropped to 0.2692, and the ASR value also exhibited a significant decline to 0.2809. Both values are less than the threshold; therefore, they fall under the unsuccessful cases. Additionally, with a 3% FP attack, the server accuracy decreased to 0.0491 and the ASR also decreased to 0.0575, which again fulfills the requirements of the unsuccessful case. Next, such cases were observed with 5%, 7%, 10%, and 15% of the feature-poisoned data, where the accuracy in all cases was less than 0.3500, which is less than the threshold value. Furthermore, the ASR value in these three cases was less than 0.36, which is again less than our threshold. These attack cases cannot fool the server and are easily detected by the system.

TABLE VI
FEATURE POISON (FP) ATTACK FOR THE SCENARIO N_{BAU1}^{CIC-FP} ON CIC DATASET

Poison	Component	Loss	Accuracy	ASR
1%	Client-1 & Client-2	0.6507 & 0.6507	-	-
1%	Server	-	0.9642	0.9628
4%	Client-1 & Client-2	0.6754 & 0.6754	-	-
4%	Server	-	0.8611	0.8616
20%	Client-1 & Client-2	0.6785 & 0.6790	-	-
20%	Server	-	0.7427	0.7763
25%	Client-1 & Client-2	0.6321 & 0.6321	-	-
25%	Server	-	0.9680	0.9671

E. Results With Feature Poisoning (FP) Attack on UNSW dataset

This section is related to scenarios with a feature-poisoning attack on the UNSW dataset. In Table VII we list the successful cases that show the impact of the FP attack on the UNSW dataset. Again, in this scenario, we regarded the poison percentages of the attack as successful, in which the server accuracy and ASR both have values greater than a threshold value, which was set to 0.40. We manipulated the values of one feature of the UNSW dataset and trained the model with 1% of the manipulated values of the most important features in the UNSW dataset. With 1% poisoned data, we observed that the accuracy of the server was 0.8195 and, more interestingly, the ASR value was 0.8232. The accuracy of the server is good,

and the ASR is also greater than the threshold value, which means that in this case, our attack worked perfectly because it is difficult to detect by the server, and our attack can act as a ghost in the system. The next case that could cross the threshold value was poisoning of the 2% and 3% values of the feature column. After training, we tested the accuracy of the server, which was above 0.84 in both cases. Additionally, the ASR for the 2% and 3% poisoning cases was also noted to be above 0.81, as shown in Table VII which is a clear indication of successful attack cases because both values are higher than the threshold in both cases. Subsequently, all the poison percentages except 4% were observed to be successful cases. The accuracy and ASR values at 5%, 7%, 10%, 15%, 20%, and 25% poisoning percentages are greater than the threshold; therefore, if we poison the UNSW dataset with these percentages, we can fool the system.

There was only one unsuccessful case in this scenario in which the poison percentage was set to 4%, and when the model was trained and tested after manipulating the values according to this percentage of the feature column, we obtained a server accuracy value of 0.0952, while the ASR value was also observed to be 0.1089.

TABLE VII
FEATURE POISON ATTACK FOR THE SCENARIO $NBAU1^{UNSW-FP}$ ON
UNSW DATASET

Poison	Component	Loss	Acc	ASR
1%	Client-1 & Client-2	0.6779 & 0.6780	-	-
1%	Server	-	0.8195	0.8231
2%	Client-1 & Client-2	0.6769 & 0.677	-	-
2%	Server	-	0.8527	0.8722
3%	Client-1 & Client-2	0.672 & 0.672	-	-
3%	Server	-	0.8433	0.8194
5%	Client-1 & Client-2	0.6729 & 0.6728	-	-
5%	Server	-	0.8620	0.8491
7%	Client-1 & Client-2	0.6691 & 0.6690	-	-
7%	Server	-	0.9017	0.9009
10%	Client-1 & Client-2	0.6718 & 0.6723	-	-
10%	Server	-	0.8725	0.8944
15%	Client-1 & Client-2	0.6454 & 0.6462	-	-
15%	Server	-	0.9066	0.9086
20%	Client-1 & Client-2	0.6939 & 0.6939	-	-
20%	Server	-	0.4682	0.4824
25%	Client-1 & Client-2	0.6640 & 0.6649	-	-
25%	Server	-	0.9018	0.9064

V. CONCLUSIONS AND FUTURE WORKS

In this study, we explored the vulnerability of FL models to data poisoning attacks in the computer network domain. For this purpose, we carried out two data poisoning attacks, LF and FP, on two prominent datasets, CIC and UNSW, which are part of numerous studies in computer networks. We divided the experiments into four different scenarios for both attacks and both datasets, and reported the results in different tables in Section IV. Our findings provide critical insights into the susceptibility of FL models to adversarial attack. We

determined that the LF attack is not an effective approach to fool the server because the accuracy drops significantly, and the system can notice it immediately and raise a critical alert. However, the FP attack showed significantly more effective results and was able to fool the server in most cases in both scenarios. Feature importance analysis revealed that a few features in the dataset can significantly influence the decision-making process of the model. If we successfully find these important features and exploit them, then these vital features in FP attacks require enhanced feature-protection mechanisms.

In future work, enhanced advanced defense strategies can be proposed to defend the FL against data-poisoning attacks. Studies can focus on feature obfuscation and protection strategies that can reduce the influence of feature-poisoning attacks, including feature-level encryption and feature-perturbation techniques. Moreover, researchers should delve into more sophisticated adversarial strategies, including transfer attacks between models, evasion attacks, and adversarial federated learning, to bolster defense against these evolving threats. By identifying attack vectors and their critical impacts, we can establish a framework for developing more resilient and enduring FL systems and enhancing their capability to face challenging threats in a dynamic cybersecurity environment. In addition, we experimented with two clients; however, in future studies, it can be extended to N clients, where the value of N is greater than 2, and more interesting results can be expected.

REFERENCES

- [1] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data," *Journal of Big Data*, vol. 7, no. 1, p. 104, 2020.
- [2] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.
- [3] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," 2021.
- [4] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 267–284.
- [5] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," 2018.
- [6] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," 2019.
- [7] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "Loadaboost: loss-based adaboost federated machine learning with reduced computational complexity on iid and non-iid intensive care data," 2020.
- [8] P. Liu, X. Xu, and W. Wang, "Threats, attacks and defenses to federated learning: Issues, taxonomy and perspectives," *Cybersecurity*, vol. 5, no. 4, pp. 1–4, 2022. [Online]. Available: <https://doi.org/10.1186/s42400-021-00105-6>
- [9] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022.
- [10] Y. Shi and Y. E. Sagduyu, "Evasion and causative attacks with adversarial deep learning," in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, 2017, pp. 243–248.
- [11] T. Luo, Y. Zhong, and S. Khoo, "Exploreadv: Towards exploratory attack for neural networks," 2023.
- [12] Y. Shi and Y. Sagduyu, "Evasion and causative attacks with adversarial deep learning," 10 2017, pp. 243–248.
- [13] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," *Machine Learning*, vol. 81, no. 2, pp. 121–148, 2010. [Online]. Available: <https://doi.org/10.1007/s10994-010-5188-5>

- [14] C. Online. (2023) How data poisoning attacks corrupt machine learning models. [Online]. Available: <https://www.csoonline.com/article/570555/how-data-poisoning-attacks-corrupt-machine-learning-models.html>
- [15] X. Cao and N. Z. Gong, "Mpaf: Model poisoning attacks to federated learning based on fake clients," 2022.
- [16] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," 2017.
- [17] 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 *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, may 2019. [Online]. Available: <https://doi.org/10.1109/SP.2019.00065>
- [18] A. Paudice, L. Muñoz-González, and E. C. Lupu, "Label sanitization against label flipping poisoning attacks," 2018.
- [19] U. M. L. Repository, "Iris," UCI Machine Learning Repository, 1988, DOI: <https://doi.org/10.24432/C56C76>.
- [20] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [21] R. E. F. G. Hopkins, Mark and J. Suermondt, "Spambase," UCI Machine Learning Repository, 1999, DOI: <https://doi.org/10.24432/C53G6X>.
- [22] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrasamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," 2017.
- [23] J. Feng, Q.-Z. Cai, and Z.-H. Zhou, "Learning to confuse: Generating training time adversarial data with auto-encoder," 2019.
- [24] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)," [Online]. Available: <http://www.cs.toronto.edu/~kriz/cifar.html>
- [25] A. Raza, S. Li, K.-P. Tran, and L. Koehl, "Using anomaly detection to detect poisoning attacks in federated learning applications," 2023.
- [26] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," 2020.
- [27] E. Hallaji, R. Razavi-Far, M. Saif, and E. Herrera-Viedma, "Label noise analysis meets adversarial training: A defense against label poisoning in federated learning," *Knowledge-Based Systems*, vol. 266, p. 110384, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S095070512300134X>
- [28] R. Alshammari and A. N. Zincir-Heywood, "Can encrypted traffic be identified without port numbers, ip addresses and payload inspection?" *Computer Networks*, vol. 55, no. 6, pp. 1326–1350, 2011. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128610003695>
- [29] C. Yang, Q. Wu, H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," 2017.
- [30] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," 2017.
- [31] T. D. Nguyen, P. Rieger, M. Miettinen, and A.-R. Sadeghi, "Poisoning attacks on federated learning-based iot intrusion detection system," 2020. [Online]. Available: <https://api.semanticscholar.org/CorpusID:216086694>
- [32] M.-O. S. N. Wolberg, William and W. Street, "Breast Cancer Wisconsin (Diagnostic)," UCI Machine Learning Repository, 1995, DOI: <https://doi.org/10.24432/C5DW2B>.
- [33] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, "Automated dynamic analysis of ransomware: Benefits, limitations and use for detection," 2016.
- [34] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [35] G. Moody and R. Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.
- [36] A. D. G. A. O. L. Reyes-Ortiz, Jorge and X. Parra, "Human Activity Recognition Using Smartphones," UCI Machine Learning Repository, 2012, DOI: <https://doi.org/10.24432/C54S4K>.
- [37] E. Nowroozi, M. Mohammadi, P. Golmohammadi, Y. Mekdad, M. Conti, and S. Uluagac, "Resisting deep learning models against adversarial attack transferability via feature randomization," 2022.
- [38] X.-H. Nguyen and K.-H. Le, "Robust detection of unknown dos/ddos attacks in iot networks using a hybrid learning model," *Internet of Things*, vol. 23, p. 100851, 2023.
- [39] nfstream, "nfstream - network traffic and flow analysis," <https://www.nfstream.org/>, 2023, accessed: 21-09-2023.
- [40] Pytorch documentation. Accessed on September 10, 2023. [Online]. Available: <https://pytorch.org/get-started/locally/>
- [41] I. H. Ehsan Nowroozi, "Federated learning under attack: Exposing vulnerabilities through data poisoning attacks in computer networks," https://github.com/ehsanowroozi/FederatedLearning_Poison_LF_FP, 2023, accessed: November 7, 2023.
- [42] E. I. Polat. (2020) Federated learning: A simple implementation of fedavg (federated averaging) with pytorch. Accessed: September 27, 2023. [Online]. Available: <https://tinyurl.com/2kvkbe98>
- [43] S. learn contributors, "Feature importances with a forest of trees," Scikit-learn documentation, 2023. [Online]. Available: https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html



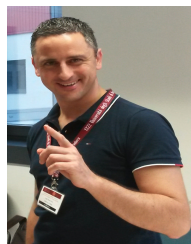
security, AI for Security, Adversarial Machine Learning, and Multimedia Forensics and Security.



essential skill for identifying and mitigating security vulnerabilities in digital systems.



Rahim Taheri (Member, IEEE) received his Ph.D. degree in Information Technology from Shiraz University of Technology, Iran, in 2020. Now he is a Lecturer in Cyber Security and Forensics at the University of Portsmouth, UK. Before joining the University of Portsmouth, he was a post-doctoral research associate at King's Communications, Learning, and Information Processing (Kclip) Lab, King's College London, UK. His main research interests include machine learning applications in security, adversarial machine learning, and federated learning.



Mauro Conti (Fellow, IEEE) received the Ph.D. degree from the Sapienza University of Rome, Italy, in 2009. He is a Full Professor at the University of Padua, Italy. He is also affiliated with TU Delft and the University of Washington, Seattle. His research in the area of Security and Privacy is also funded by companies, including Cisco, Intel, and Huawei. He published more than 500 papers in topmost international peer-reviewed journals and conferences. He is the Editor-in-Chief for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and has been an Associate Editor for several journals, including IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. He is a Fellow of YAE and a Senior Member of ACM.