ABSTRACT  Nowadays, Federated Learning has widely been adopted for data security in the Industrial IoTs. With Federated Learning, local Industrial IoTs devices download the current machine learning model and update it on their own local Industrial IoTs devices. Then, local Industrial IoTs devices transmit these locally trained models back to the Industrial Server. The Industrial Server aggregates all the locally trained models into a single consolidated and enhanced global model. On one side, Federated Learning secures the data; on the other side, Federated Learning itself is vulnerable to one subtle yet severe attack: the model poisoning attack. Model poisoning attack is difficult to detect, especially in Industrial IoTs applications, for two reasons: a) neither the Industrial Server nor the local Industrial IoTs devices in Federated Learning is capable of identifying poisoned local models, and b) every iteration of Federated Learning consists of many Industrial IoTs devices, and therefore, verification of every single device is computationally expensive. Thus, this study proposes an effective and efficient framework for deTectIon of Model Poisoning Attacks usiNg AccuracY (TIMPANY). TIMPANY is the first detection framework for the model poisoning attack that utilizes accuracy as a detection measure. We performed theoretical analysis of TIMPANY with other detection solutions (for model poisoning attack) concerning communication and computational efficiency, security, and detection accuracy. Our thorough theoretical comparative analysis showed that TIMPANY efficiently addresses these open research challenges that previous studies failed to address. In our thorough experimental analysis, error analysis from the first iteration shows that TIMPANY results in 0% error, leading to a True Positive Rate and accuracy of 100% with 0% False Positive Rate. Thus, TIMPANY outperformed some of the existing detection solutions for model poisoning attacks against Federated Learning. We conclude that TIMPANY is effective and efficient against model poisoning attacks in Federated Learning, even for resource-constrained Industrial IoTs devices widely used in various industrial applications.

INDEX TERMS  Federated Learning, Model poisoning attack, Detection framework, Accuracy, Industrial IoTs.
Federated Learning has shown promising results in various Industrial IoTs applications because:

1) it enables collaborative learning while keeping data on personal devices; hence reducing the cost of storage memory at Industrial IoTs server, and
2) it provides security to personal data which can be leaked while transmission [4].

However, on the other hand, Federated Learning also suffers and lacks in various aspects [5]–[7]. For example:

- **expensive communication**: Federated Learning involves a massive number of Industrial IoTs devices because every device performs training according to the local computation capacity. Hence it slows down the network leading to costly communication.
- **system heterogeneity**: An IoT devices interact with several Industrial IoTs devices comprised of different computation, storage, network capabilities. These characteristics of Industrial IoTs devices may cause strugglers and faults in the network. This slows down the updating process of the global model.
- **statistical heterogeneity**: In the Federated Learning network, there is numerous Industrial IoTs devices that generate and collect data in a non-identically distributed manner. Moreover, the number of data points vary from device to device in the Federated Learning network. This paradigm of data generation violates the independent and identically distributed (i.i.d) Hence it may increase strugglers and, adds complexity to modelling, analysis and evaluation, and
- **model poisoning**: Though the approach of Federated Learning introduces privacy and protection to the Industrial IoTs’ data by not being transmitted to the Industrial IoTs server. Federated Learning opens new security vulnerability which is yet to be solved, i.e., model poisoning.

Several studies have proposed to overcome the challenges of expensive communication, system heterogeneity, and statistical heterogeneity. For example, in [8], [10], [28], the authors have proposed solutions to improve communication cost and energy consumption. On the other side, very few studies have focused on the detection or mitigation of model poisoning attacks. Moreover, those proposed studies either gave rise to new security challenges or were not viable for various Industrial IoTs applications [18]–[23], as discussed in Section II.

Therefore, in this study, we particularly tailored a detection framework for model poisoning attacks against a Federated Learning, which is viable for the industrial environment. Our proposed framework is a novel privacy-preserving framework, TIMPANY, that utilizes accuracy as a detection measure.

The acronyms and abbreviations used throughout this study are mentioned in Table 1.

### A. CONTRIBUTIONS

The main contributions of this study can be summarized as follows.

1) The proposed framework, TIMPANY, analyzes the local model weights for model poisoning attacks by itself at the server side as clients have to send their respective local model weights directly to the server. Thus, clients’ summarized weights are secure from any other client or party in the FL. Hence, TIMPANY maintains the privacy and security of every single client within the FL setup.

2) TIMPANY follows vanilla FL and analyzes local model weights at the server side. Thus, TIMPANY does not require additional computation and communication cost to detect poisoned local model weights. Therefore, TIMPANY is a viable framework for any environment, even for a resource-constraint IoT-based industrial environment.

3) TIMPANY does not rely on clients of the FL for analysis and evaluation of local model weights. Therefore, TIMPANY does not misevaluate the local model weights. As a result, TIMPANY has significantly high TPR and accuracy of 100% while having noticeably low FPR of 0%.

The rest of the paper is organized as follows: Section II describes some related work and some of the existing research gaps. Section III presents our proposed solution along with its detailed methodology. Section IV highlights the experiments and evaluations of our proposed solution. Finally, Section V concludes this paper along with the future work and directions.

### II. RELATED WORK

In [11]–[13], the authors have devised model poisoning attacks employing a boosting mechanism and altering optimization strategy for stealthy model poisoning. In [14] and, [15], the authors conducted a thorough study involving recent advances, open challenges and problems posed by FL. In [16], the authors have used model poisoning attacks to introduce backdoor attacks in FL. Their results showed that attacks are possible with only 0.01% access to the devices. In [17], the authors conducted a study on weight poisoning attacks (also known as model poisoning attacks) on pre-trained models. The authors have discussed the defence and illustrated the effectiveness of these for exposing the backdoors. In [18], the authors proposed local model poisoning against Byzantine-Robust FL. In [19], authors have demonstrated model poisoning attacks through data poisoning, specifically, label flipping poisoning attacks with which authors can also negatively impact the global model.

In [18], the authors have also discussed generalized two data poisoning defences against their proposed attacks based on the largest error loss and negative impact. However, these defences have limited success. The authors, in [19], presented the dimensionality reduction-based defence mechanism ca-
TABLE 1. Acronyms and Abbreviations

| Acronyms and Abbreviations | Description |
|-----------------------------|-------------|
| CL Central Limit            |             |
| FL Federated Learning       |             |
| FN False Negative           |             |
| FP False Positive           |             |
| FPR False Positive Rate     |             |
| FP  True Negative           |             |
| TP  True Positive           |             |
| TPR True Positive Rate      |             |
| IS Industrial Server        |             |
| LCL Lower Control Limit     |             |
| UCL Upper Control Limit     |             |
| IoT Internet of Things      |             |

In summary, those proposed detection solutions lack in multiple aspects, thus leading to various main research gaps and challenges. The first and foremost aspect is a need of such a verification process for poisoned local model weights which should be efficient and effective, i.e., the process consumes less computation and communication cost while maintaining the attacks detection accuracy (no/less misevaluated as legitimate local model weights). In [22], the authors proposed to utilize gradients instead of the model’s parameter for filtering Byzantine clients. The solution is analyzed based on a random selection of training and validation local datasets. Thus the solution is not viable for the FL environment because the experimental setup negated the concept of actual FL setup. Moreover, the solution consumes additional computational time for the local model weights evaluation. Furthermore, the solution lacks in defining a discriminating decision boundary. Therefore, the solution also causes misevaluation of local model weights. Also, in [23], the authors presented a stochastic quantization-based detection framework for Byzantine clients. Their proposed solution puts additional computation both on a server as well as the client side in terms of clients’ selection and computation of pairwise distances.

In summary, those proposed detection solutions lack in multiple aspects, thus leading to various main research gaps and challenges. The first and foremost aspect is a need of such a verification process for poisoned local model weights which should be efficient and effective, i.e., the process consumes less computation and communication cost while maintaining the attacks detection accuracy (no/less misevaluated as legitimate local model weights). In [22], the authors proposed to utilize gradients instead of the model’s parameter for filtering Byzantine clients. The solution is analyzed based on a random selection of training and validation local datasets. Thus the solution is not viable for the FL environment because the experimental setup negated the concept of actual FL setup. Moreover, the solution consumes additional computational time for the local model weights evaluation. Furthermore, the solution lacks in defining a discriminating decision boundary. Therefore, the solution also causes misevaluation of local model weights. Also, in [23], the authors presented a stochastic quantization-based detection framework for Byzantine clients. Their proposed solution puts additional computation both on a server as well as the client side in terms of clients’ selection and computation of pairwise distances.

In summary, those proposed detection solutions lack in multiple aspects, thus leading to various main research gaps and challenges. The first and foremost aspect is a need of such a verification process for poisoned local model weights which should be efficient and effective, i.e., the process consumes less computation and communication cost while maintaining the attacks detection accuracy (no/less misevaluated as legitimate local model weights). In [22], the authors proposed to utilize gradients instead of the model’s parameter for filtering Byzantine clients. The solution is analyzed based on a random selection of training and validation local datasets. Thus the solution is not viable for the FL environment because the experimental setup negated the concept of actual FL setup. Moreover, the solution consumes additional computational time for the local model weights evaluation. Furthermore, the solution lacks in defining a discriminating decision boundary. Therefore, the solution also causes misevaluation of local model weights. Also, in [23], the authors presented a stochastic quantization-based detection framework for Byzantine clients. Their proposed solution puts additional computation both on a server as well as the client side in terms of clients’ selection and computation of pairwise distances.
III. METHODOLOGY

In this study, we have proposed TIMPANY, a detection framework for model poisoning attack as shown in Figure 1. It is the first framework that utilizes the accuracies of the local model weights to identify and detect model poisoning attacks in FL. Moreover, TIMPANY effectively overcomes all the research gaps and challenges posed by previous studies, as highlighted in Section II. Also, the working mechanism of TIMPANY enables TIMPANY a viable model poisoning detection framework for various environments, even for resource-constraint applications.

Figure 2 illustrates the functionalities and workflow of the entire detection process in a module diagram form of Figure 1. Following steps further describe the mechanism of the entire detection process.
STEP 1: INDUSTRY
TIMPANY follows the classical FL setup, i.e., initially a server would train a global model (neural network) with the initial training dataset as

\[ \text{Global\_Model} = \text{weights}_{n-1} \times \text{global\_train\_data} \]  

where,
- \text{weights}_{n-1} = \text{initial weights of the neural network},
- \text{train\_data} = \text{initial training data given to the neural network},

and then deploy the trained global model to the \( m \) number of the local devices of the participants in the FL setup, i.e., \( FL = \sum_{j}^{m} \). Then, the participants utilize their personal streaming datasets to train the deployed global model on their own local devices. Mathematically we can write it as,

\[ w_1 = \text{Local\_Model}(\text{weights}_{n-1} \times \text{local\_data}_1), \]
\[ w_2 = \text{Local\_Model}(\text{weights}_{n-1} \times \text{local\_data}_2), \]
\[ w_3 = \text{Local\_Model}(\text{weights}_{n-1} \times \text{local\_data}_3), \]
\[ \vdots \]
\[ w_m = \text{Local\_Model}(\text{weights}_{n-1} \times \text{local\_data}_m) \]  

where \( w_m \) are the updated weights of each participant in FL setup. \( \text{Local\_Model} \) is the global model which is now deployed on the participant(s) device, and \( \text{local\_data}_m \) is each participant’s data which is utilized by each participant by itself to train the local model on their own devices.

STEP 2: SERVER
Once the participants are done with local training process, the participants summarize their local model weights and send those summaries individually and independently to the server, i.e., \( \Delta w_1, \Delta w_2, \Delta w_3, \ldots, \Delta w_m \).

STEP 3: TIMPANY
Then, the server employs TIMPANY to identify and detect poisoned local model weights.

**Step 3a: Global Model with local model weight(s)**
TIMPANY evaluates each local model weight by setting up each local model weight as new parameter(s) for global model.

**Step 3b: Initial Testing Dataset**
To test the global model’s accuracy with local model weights, TIMPANY utilizes the initial global model’s testing dataset, i.e.,

\[ \text{Global\_Model}_m = \Delta w_m \times \text{global\_test\_data} \]  

where,
- \text{global\_test\_data} = \text{the initial testing dataset used by the initial global model for testing}.

**Step 3c: Evaluation of accuracy using quality control charts**

Thereafter, using quality control charts\(^1\), TIMPANY sets the T based on accuracies computed using local model weights. To compute the quality control charts, TIMPANY first computes \( \bar{X} = \frac{\sum_{i=1}^{n} \text{acc}_i}{n} \), and \( \sigma = \sqrt{\frac{\sum_{i=1}^{n} (\text{acc}_i - \bar{X})^2}{n}} \), where \( \text{acc}_i \) = accuracies obtained from the local model weights. Then TIMPANY computes \( CL = \bar{X} + \alpha * \sigma \) and \( LCL = \bar{X} - \alpha * \sigma \) where \( \alpha \) = range of limits, i.e., 1, 2, and 3.

At last, TIMPANY discards those local model weights (as poisoned local model weights) based on their accuracies which are below the computed \( T \), i.e.,

\[ \Delta w_m = \begin{cases} \text{discard}, & \text{if acc} \geq T \\ \text{keep}, & \text{if acc} < T \end{cases} \]

| Algorithm 1: Algorithm for TIMPANY |
|------------------------------------|
| **Input:** \( \text{Global\_Model}_t \), weight = \( [\Delta w_1, \ldots, \Delta w_m] \)  |
| **Output:** \( \text{Global\_Model}_{t+1} \)  |
| **Set variable:** ind, sum, aggregate, count, temp = 0  |
| **Set variable:** acc = []  |
| **procedure** Update\_Global\_Model (weight)  |
| for ind in len(weight) do  |
| temp = Global\_Model_t  |
| Global\_Model_{ind} = temp.predict(test\_data)  |
| accuracy[ind] = Global\_Model_{ind}.cross_val_score()  |
| ind = ind + 1  |
| end for  |
| for a in len(acc) do  |
| sum = sum + acc[a]  |
| end for  |
| CL = sum / len(acc)  |
| std\_dev = np.std(acc)  |
| UCL = CL + (\alpha * std\_dev)  |
| LCL = CL - (\alpha * std\_dev)  |
| \( T \) (determined through graph)  |
| for b in len(acc) do  |
| if T <= acc[b] then  |
| acc[b] = 0  |
| else  |
| aggregate = acc[b]  |
| count = count + 1  |
| end if  |
| end for  |
| aggregate = aggregate / count  |
| return Global\_Model_{t+1}(aggregate)  |
| **end procedure**  |

Algorithm 1 shows a complete working of TIMPANY for detecting model poisoning attacks against FL in IIoT applications. The workflow of the TIMPANY has shown that TIMPANY has addressed the open research challenges of the studies mentioned in Section II, i.e., TIMPANY verifies and detects the model poisoning attacks using accuracy at

\(^1\)A graphical representation to determine the level of variation.
the server side in the FL setup. Therefore, TIMPANY is completely independent of client-based verification process. Eventually, TIMPANY preserves the security and privacy of participants in FL setup while maintaining the detection accuracy. Moreover, TIMPANY does not consume any additional communication cost between server and participants in IIoT applications. Also, TIMPANY evaluates the local model weights immediately after receiving the local model weights without consuming any additional computation cost both at the client and server side. In addition to this, TIMPANY computes optimal T value for every round of FL, thus leading to no miscalculation of local model weights. These desirable features of the novel framework, TIMPANY, give TIMPANY the precedence over some of the existing detection solutions.

**STEP 4: GLOBAL MODEL**

In this last step, the global model is updated using the aggregation of all the trusted local model weights (selected from Step 3c).

For the second and onwards rounds or iterations in the FL setup of the IIoT applications, the whole detection process will be repeated in the similar manner as that of first round.

**IV. EXPERIMENT**

There are various potential applications of FL in the IIoT environment. Researchers have coupled FL with different other approaches to improve the performance of the industries in the IIoT environment [24], and [25]. For example, in [26]–[29], the authors have demonstrated that FL for the detection of device failure and DDoS attacks in IIoT applications. However, FL itself is vulnerable to model poisoning attacks which can adversely affect the performance of the smart industries. Limited work is present for the security of FL, particularly for IIoT applications. Therefore, to demonstrate the application of our proposed framework, TIMPANY, we have considered a sample scenario of an industrial environment.

**A. SIMULATION ENVIRONMENT**

We developed our proposed framework using Python version 3.8.0 under the LINUX Ubuntu 16.04 environment. First, we set up multiple client-server architectures using Python libraries including socket, pickle, time, and threading. Further, to develop and build distributed learning, we have used PyGAD.

**B. DATA PREPARATION**

To demonstrate the concrete realization of the proposed framework, TIMPANY, we have used the CIFAR-100 dataset. CIFAR-100 dataset is one of the labelled subsets from 80 million tiny images. CIFAR-100 dataset is composed of 100 classes and, each class consists of 600 images. Furthermore, each class hold 500 training images and 100 testing images.

For our experimentation, we divided the entire CIFAR-100 dataset into six datasets(subsets). Each sub-dataset consists of 9,000 training images and 1,000 testing images having all the 100 classes. There can be two types of cases through which the malicious participant can generate and prepare poisoned local model weights. For the first type of case, the malicious participant trains the deployed model on a correct dataset and then modifies the local model weights to make those local model weights corrupted weights. In the second type of case, the malicious participant trains the global model on the poisoned dataset. Thus, resulting in poisoned local model weights. Therefore, to cater latter possibility, we introduced Gaussian and Salt&Pepper noises to the sub dataset of Participant F. Whereas, for the remaining cases, we utilized the sub-datasets directly(unchanged) for Participants A, B, C, D, and E. The details on the considered cases in this study, characteristics of each participant, and how each participant communicate with the server are mentioned in Section IV-C.

**C. EXPERIMENTAL SETUP**

To demonstrate the applicability of our proposed framework, TIMPANY, we have considered a scenario, where we have conducted several experiments based on all possible model poisoning attack cases as shown in Figure 3. In this sample scenario, we take into account the following cases.

- Case I - Legitimate participant with secure communication channel
- Case II - Legitimate participant with compromised communication channel
- Case III - Legitimate participant with physically hijacked local device
- Case IV - Malicious participant who intentionally transmits poisoned local model weights

For the first case, we sent the global model to Participant A’s and E’s devices. Participant A and E then use their datasets (prepared and explained in the Section IV-B) to train their local models, summarize their local model weights and send them to the server.

For the second case, we deployed the global model to Participant B’s device. Similar to Participant A, Participant B also utilizes its dataset to train its local model and summarizes its local model weights. However, while sending the summarized local model weights to the server, there was a man-in-the-middle attack, i.e., the attacker was tapping the communication channel between Participant B and the server. Hence, during the communication, the attacker manipulates the summarized local model weights (gradients) and forwards the poisoned local model weights to the server.

In the third case, we also deployed the global model to Participant C’s device. However, in this case, the attacker has physical access to Participant C’s device. Therefore, when Participant C trains its local model with its dataset and summarizes its local model weights, the attacker modifies the summarized local model weights on Participant C’s device. As a result, even if the communication channel between the server and Participant C is secure, the summarized local model weights sent to the server by Participant C are poisoned local model weights.
Finally, in the fourth case, we deployed the global model to Participant D’s and Participant F’s devices for training. Since Participant D and Participant F are themselves the malicious participants. Therefore, Participant D and Participant F intentionally send the poisoned local model weights to the server. Thus, the aim of Participant D and Participant F is to corrupt the global model and cause misclassifications.

**D. ASSUMPTIONS**

In this study, we have assumed that the participants in the FL setup have constant communication and connection with the server. Also, this study primarily focused on the model poisoning attacks against FL through the detection framework. Therefore, problems of FL, e.g., system heterogeneity and statistical heterogeneity, are out of the scope of this study.

**E. EVALUATION METRICS**

With the advent of sophisticated cyber attacks, it is becoming essential to quantify the detection accuracy of the cyber security systems against sophisticated cyber attacks. Therefore, to quantify the TIMPANY’s accuracy in detecting the model poisoning attacks, we employed Percent Error (also known as a Percentage Error [30]). Percent Error is a type of error used to indicate the percentage of error in the analysis process. Percent Error is a difference between the estimated values and the actual values divided by the actual values as

\[
\text{Percentage Error} (\delta) = \frac{|v_E - v_A|}{v_A} * 100
\]

where,

- \(v_A\) = no. of actual poisoned local model weights
- \(v_E\) = no. of estimated poisoned local model weights by TIMPANY

To evaluate and demonstrate the detection accuracy of TIMPANY, we have also computed the values of the confusion matrix as shown in Table 2.

**TABLE 2. Confusion Matrix**

| Actual: Normal | Estimated: Normal | Estimated: Poison |
|----------------|-------------------|------------------|
| TN = 2         |                   |                  |
| FP = 0         |                   |                  |

where,

- TP = correctly identified poisoned local model weights

\[\text{Percentage Error} (\delta) = \frac{|v_E - v_A|}{v_A} * 100\]
TN = correctly identified normal local model weights
FP = incorrectly identified poisoned local model weights
FN = incorrectly identified normal local model weights

Using the values of the confusion matrix, we then calculated TPR (also known as Sensitivity, to measure the percentage of correctly identified poisoned local model weights), FPR (also known as Fall-Out, to measure the percentage of incorrectly identified poisoned local model weights), Accuracy (to measure the percentage of correctly identified local model weights), and F1-score (also known as F1-measure, to measure the performance of the system).

where,

\[
TPR = \frac{TP}{TP + FN}
\]

\[
FPR = \frac{FP}{FP + TN}
\]

\[
Accuracy = \frac{TP + TN}{TP + FP + TN + FN}.
\]

F. TIMPANY ANALYSIS

We evaluated TIMPANY both using theoretical analysis and experimental analysis. With theoretical analysis, we have analysed the concepts and structure of TIMPANY against some of the other existing solutions. On the other hand, in the experimental analysis, we have evaluated the performance and detection accuracy of TIMPANY with some of the proposed solutions.

1) Theoretical Analysis

We theoretically analyzed and compared our proposed TIMPANY with some of the other existing defences against model poisoning attacks solutions. Below is the detailed analysis and comparison, whereas the summarized analysis is mentioned in Table 6.

1) Client-based verification:

Since TIMPANY analyses accuracies, TIMPANY is capable of analyzing weights based on the performance, unlike [20] and [23], which employed client-based verification to compute pairwise distances to identify the dissimilarity among clients’ local model weights.

2) Additional communication:

TIMPANY performs the detection procedure on the server side. Therefore, the TIMPANY does not cause additional communication cost, unlike of [20] and [23], which required clients to compute and exchange various information within the network and with the server for the identification of poisoned weights. Thus this leads to additional communication and computation between server and clients.

3) Additional computation:

In [18], [22], [19] and [21], the proposed defences consumed additional computation time to compute gradients, high dimensional update vectors, Principal Component Analysis, and reconstruction errors, respectively. The solutions proposed in [20] and [23] are required to compute dissimilarity and related information through clients. Thus incurred additional computation cost.

However, TIMPANY evaluates and identifies the poisoned local model weights immediately after receiving the local model weights from the participants in the FL setup. Hence, TIMPANY instantly determines the poisoned local model weights without any excessive additional computation.

4) Security and privacy:

In contrast, to [20] and [23], which involve clients in the identification and verification procedure of the poison local model weights, TIMPANY, evaluates the local model weights without indulging the clients in the evaluation procedure at the server side. As a result, TIMPANY provides complete security and privacy to the clients, i.e., no information gets disclosed to another client in the FL setup.

5) Misevaluation of weights:

The solution presented in [18] computes the largest error rate from the local model weights. However, this solution may incorrectly accept poisoned local model weights as it only looks for the largest error loss. Thus, this results in the misevaluation of the local model weights.

Since in FL, each participant can access the global model along with all the parameters, and the attackers can make minor changes in the existing parameters while keeping most of the parameters unchanged, the approach of looking for unique characteristics between poisoned and legitimate local model weights may cause misevaluation of local model weights [19].

Next, the defence mechanism of [21] employs the mean value of reconstruction errors as a T value to distinguish between legitimate and poisoned local model weights. This mechanism may also lead to severe misevaluations. Let us assume a sample scenario for [21], where the number of legitimate local model weights is more than the number of poisoned local model weights.

Then, the mean value of reconstruction errors would shift towards the legitimate local model weights. Thus, poisoned local model weights can be miscalculated as legitimate ones and vice versa.

In [20] and [23], participants (specifically attackers) can miscalculate the dissimilarity values and related information. Thus, local model weights can be miscalculated and severely corrupt the global model.

On the contrary to these existing defences, TIMPANY evaluates the local model weights at the server side using the accuracies of the local model weights.
TABLE 3. Comparison with some of the existing state-of-the-art model poisoning attack detection solutions

| Main Drawbacks                        | Defences against model poisoning attacks |
|---------------------------------------|------------------------------------------|
| Requires client-based verification     | [18] X                                   |
| Requires additional communication     | [19] X                                   |
| Requires additional computation       | [20] X                                   |
| Lacks privacy and security            | [21] ✓                                   |
| Misevaluation of weights              | [22] ✓                                   |
| Lacks optimized T/boundary value      | TIMPANY ✓                                |

In conclusion, from the first iteration, local model weights received by the server from Participant B and Participant C are poisoned, i.e., the resultant accuracies of Participant B and Participant C on the initial testing dataset of the global model should be low. Hence, the accuracies of Participant B and Participant C are less than the computed optimal T, which leads to rejection of Participant B and Participant C local model weights.

In conclusion, from the first iteration, local model weights from only Participant A and E are selected as trusted local model weights. These chosen model weights are then aggregated and used to update the initial global model from the

However, in the assumed sample scenario, the local model weights received by the server from Participant B and Participant F local model weights are discarded as poisoned local model weights.

FIGURE 4. The graph of computed optimal T value for the first iteration of FL. The computed optimal T value is 0.69. The accuracies of Participant A and Participant E are the only accuracies which are greater than the computed optimal T. On the other hand, the accuracies of Participant B, C, D, and F are less than the computed optimal T. Thus, from the first iteration of FL, only Participant A and Participant E local model weights are selected as trusted local model weights and are utilized to update the global model. However, Participant B, C, D, and F local model weights are discarded as poisoned local model weights.

FIGURE 4. The graph of computed optimal T value for the first iteration of FL. The computed optimal T value is 0.69. The accuracies of Participant A and Participant E are the only accuracies which are greater than the computed optimal T. On the other hand, the accuracies of Participant B, C, D, and F are less than the computed optimal T. Thus, from the first iteration of FL, only Participant A and Participant E local model weights are selected as trusted local model weights and are utilized to update the global model. However, Participant B, C, D, and F local model weights are discarded as poisoned local model weights.

2) Experimental Analysis

From Figure 4, we have computed the CL, LCL and UCL, for obtaining the optimal T value, using the formulas mentioned in Section III. The computed optimal T value is 0.69 for the first iteration of the FL setup. According to the assumed sample scenario, Participant A and Participant E are legitimate participants and should have good accuracies on the initial testing dataset of the global model. Hence, the accuracies from Participant A and Participant E positively contribute towards the improvement of the global model. Therefore, it is evident that the accuracies of Participant A and Participant E are greater than the computed optimal T value and kept for the aggregation and updation of the global model.
first iteration. The experiments performed and demonstrated in this study are for the first iteration of FL. For all iterations of FL, the same process follows, i.e., connection with the random participants, collection of local model weights, model poisoning attacks detection through TIMPANY, and updation of a global model.

After experimentation, we performed the error analysis of the TIMPANY’s accuracy. Tables 4 showed the error analysis. The computed Percentage Error from the first iteration of FL using TIMPANY is 0%. In other words, TIMPANY is effectively detecting and mitigating model poisoning attacks against FL without making misevaluation of the local model weights.

**TABLE 4.** Percentage Error Analysis of TIMPANY

| Error Analysis of TIMPANY’s Accuracy |   |
|-------------------------------------|--|
| Actual poisoned local model weights (\(\nu_A\)) | 4 |
| Estimated poisoned local model weights by TIMPANY (\(\nu_G\)) | 4 |
| Percentage Error (\(\delta\)) | 0% |

We then computed the detection accuracy of TIMPANY, as shown in Table 5. To quantify the TIMPANY’s detection accuracy against the model poisoning attacks, we have utilized TPR, FPR and accuracy as our evaluation metrics.

**TABLE 5.** Detection Accuracy Analysis of TIMPANY

| Detection Accuracy Analysis |   |
|----------------------------|---|
| TPR (Sensitivity) | 100% |
| FPR (Fall-Out) | 0% |
| Accuracy | 100% |

As shown in Table 5, it can be seen that TIMPANY achieved significantly high TPR of 100% while maintaining significantly low FPR of 0%. Also, TIMPANY has achieved an accuracy of 100%.

To evaluate and examine the performance of TIMPANY, we have compared the accuracy of TIMPANY with some of the benchmark detection solutions. To have a fair evaluation with TIMPANY, we have evaluated the accuracy achieved from the first round of FL from each compared detection solution. From Table 6, it can be seen that TIMPANY has achieved noticeably highest accuracy of 100% even from the first round of FL setup.

**TABLE 6.** TIMPANY comparison with some of the existing model poisoning attacks detection solutions

| Detection Solutions | Accuracy |
|---------------------|----------|
| Zhao, Lingchen, et al. [20] | 90% |
| Li, Suyi, et al. [21] | 70% |
| Chen, Chen, et al. [22] | 4% |
| So, J., Güler, B., & Avestimehr, A. S. [23] | 35% |
| TIMPANY (our proposed solution) | 100% |

In our thorough analysis of the solution proposed in [21], we have found that the authors in [21] have claimed that their proposed solution has achieved a system performance of 100% in terms of F1-score. Therefore, we also evaluated our TIMPANY with [21] in terms of F1-score.

**TABLE 7.** Performance Analysis of TIMPANY

| Detection Solutions | F1-score |
|---------------------|----------|
| Li, Suyi, et al. [21] | 100% |
| TIMPANY (our proposed solution) | 100% |

As shown in Table 6, both [21] and TIMPANY have achieved an F1-score of 100%. In other words, both the solutions have shown same performance. However, the solution proposed in [21] has three main limitations (e.g., consumes additional computation cost, misevaluates of local model weights and lacks optimized T value) which makes it impracticable for various IIoT applications, particularly for resource-constraint IIoT environments. On contrary, TIMPANY efficiently overcomes the limitations of [21] while maintaining an F1-score of 100%.

**V. CONCLUSION AND FUTURE DIRECTIONS**

FL has not only revolutionized classical machine learning but also provides a privacy-preserving mechanism for end-devices as well as allows decentralized learning in the industrial environment. FL has proven itself to be one of the potential solutions in IIoT applications, e.g., monitoring of defected products, controlling of pressure and gases, autonomous quality control checks, to name a few. At the same time, FL opens new security vulnerability and poses a new security threat, i.e., model poisoning attack, which can severely affect the decision-making and information classification processes in the industrial environment. These model poisoning attacks are difficult to detect because a) neither server nor participants in the FL can detect poisoned local models based on provided weights only, and b) there are many random participants in every FL iteration. Thus, verification of every participant is not viable computationally.

To address model poisoning attacks, we have proposed TIMPANY. TIMPANY is the first detection framework for model poisoning attacks which utilizes accuracy as a detection measure. We have evaluated TIMPANY both theoretically and experimentally. With theoretical analysis, we have shown that TIMPANY addresses the existing research problems which previous research studies fail to address, e.g., security and privacy maintenance, optimized T value selection, etc. Experimental analysis were conducted considering various possible cases of model poisoning attacks in the FL. Our thorough experiments showed that our proposed TIMPANY has secured a percentage error of 0%. This error analysis further resulted in 100% accuracy and TPR while securing an FPR of 0%. Thus, outperforming some of existing state-of-the-art detection solutions for model poisoning attacks. Hence, we can conclude that TIMPANY can efficiently and effectively detect the model poisoning attacks against FL.

In future work, we plan to investigate more sophisticated and automated model poisoning attacks. Furthermore, we will also plan to evaluate our TIMPANY against those so...
phisticated model poisoning attacks. Our future directions also include the enhancement of TIMPANY such that the TIMPANY can cater to other issues, e.g., system and statistical heterogeneity.

REFERENCES

[1] Pham, Quoc-Viet, Kapal Dev, Praveen Kumar Reddy Maddikunta, Thippa Reddy Gadekallu, and Thien Huynh-The. "Fusion of Federated Learning and Industrial Internet of Things: A Survey." arXiv preprint arXiv:2101.00709. 2021.

[2] Li, Li, YuXi Fan, Mike Tse, and Kuo-Yi Lin. "A review of applications in federated learning." Computers & Industrial Engineering, pp. 106854, 2020.

[3] Guo, Yinghao, Zichao Zhao, Ke He, Shiwei Lai, Junjuan Xia, and Lisheng Fan. "Efficient and flexible management for industrial Internet of Things: A federated learning approach." Computer Networks 192, pp. 108122.

[4] Yang, Qiang, Yang Liu, Tianjian Chen, and Yongxin Tong. "Federated machine learning: Concept and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 10, no. 2, pp. 1-19. 2019.

[5] Nguyen, Dinh C., Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, Dusit Niyato, and H. Vincent Poor. "Federated Learning for Industrial Internet of Things in Future Industries." arXiv preprint arXiv:2105.16559. 2021.

[6] Bouacida, Nader, and Prasant Mohapatra. "Vulnerabilities in Federated Learning." IEEE Access 9: 63229-63249. 2021.

[7] Jere, Malhar S., Tyler Farnan, and Farinaz Koushanfar. "A taxonomy of attacks on federated learning." IEEE Security & Privacy 19, no. 2, pp. 20-28. 2020.

[8] Mills, Jed, Jia Hu, and Geyong Min. "Communication-efficient federated learning for wireless edge intelligence in IoT." IEEE Internet of Things Journal 7, no. 7, pp. 5986-5994. 2019.

[9] Savazzi, Stefano, Monica Nicoli, and Vittorio Rampa. "Federated learning with cooperating devices: A consensus approach for massive IoT networks." IEEE Internet of Things Journal 7, no. 5, pp. 4641-4654. 2020.

[10] Zheng, Haifeng, Min Gao, Zhihang Chen, and Xinxin Feng. "A Distributed Hierarchical Deep Computation Model for Federated Learning in Edge Computing." IEEE Transactions on Industrial Informatics. 2021.

[11] Bhagori, Arjun Nitin, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo."Model poisoning attacks in federated learning." In Proc. Workshop Secur. Mach. Learn.(SecML) 32nd Conf. Neural Inf. Process. Syst.(NeurIPS). 2018.

[12] Bhagori, Arjun Nitin, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo."Analyzing Federated Learning through an Adversarial Lens," In International Conference on Machine Learning, pp. 634-643. PMLR, 2019.

[13] Zhou, Xingchen, Ming Xu, Yiming Wu, and Ning Zheng. "Deep Model Poisoning Attack on Federated Learning." Future Internet 13, no. 3, pp. 73. 2021.

[14] Li, Tian, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. "Federated learning: Challenges, methods, and future directions," IEEE Signal Processing Magazine 37, no. 3, pp. 50-60. 2020.

[15] McMahan, H. Brendan, "Advances and Open Problems in Federated Learning." Foundations and Trends® in Machine Learning 14, no. 1. 2021.

[16] Bagdasaryan, Eugene, Andreas Veit, Yuqing Hua, Deborah Estrin, and Vitaly Shmatikov. "How to backdoor federated learning." In International Conference on Artificial Intelligence and Statistics, pp. 2938-2948. 2020.

[17] Kurita, Keita, Paul Michel, and Graham Neubig. "Weight poisoning attacks on pre-trained models." arXiv preprint arXiv:2002.00211. 2020.

[18] Fang, Minghong, Xiaoyu Cao, Jinya Jiang, and Neil Gong, "Local model poisoning attacks to Byzantine-robust federated learning," In 29th USENIX Security Symposium (USENIX Security 20), pp. 1605-1622. 2020.

[19] Tolpegin, Vale, Stacey Trues, Mehmet Emre Gurosy, and Ling Liu, "Data Poisoning Attacks Against Federated Learning Systems," In European Symposium on Research in Computer Security, pp. 480-501. Springer, Cham, 2020.

[20] Zhao, Longchen, Shengshan Hu, Qian Wang, Jianlin Jiang, Chen Zhao, Xiangyang Luo, and Pengfei Hu, "Shielding Collaborative Learning: Mitigating Poisoning Attacks through Client-Side Detection," IEEE Transactions on Dependable and Secure Computing, 2020.

[21] Li, Suyi, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen, "Learning to Detect Malevolent Clients for Robust Federated Learning," arXiv preprint arXiv:2006.08259. 2020.

[22] Chen, Chen, Jingfeng Zhang, Anthony KH Tung, Mohan Kankanhalli, and Gang Chen, "Robust federated recommendation system," arXiv preprint arXiv:2006.08259. 2020.

[23] So, Jinhyun, Basak Guler, and A. Salman Avestimehr, "Byzantine-Resilient Secure Federated Learning," IEEE Journal on Selected Areas in Communications. 2020.

[24] Sun, Wen, Shiyu Lei, Lu Wang, Zhigang Liu, and Yan Zhang, "Adaptive Federated Learning and Digital Twin for Industrial Internet of Things." IEEE Transactions on Industrial Informatics. 2020.

[25] Zhang, Peiyang, Chao Wang, Chunxiao Jiang, and Zhi Han. "Deep Reinforcement Learning Assisted Federated Learning Algorithm for Data Management of IoT." IEEE Transactions on Industrial Informatics. 2021.

[26] Zhang, Wei-shan, Qinghua Lu, Qiu-yu Yu, Zhaotong Li, Yue Liu, Sin Kiat Lo, Shiping Chen, Xiwei Xu, and Liming Zhu. "Blockchain-based Federated Learning for Device Failure Detection in Industrial IoT." IEEE Internet of Things Journal. 2020.

[27] Hafezz, Taimur, Xia Lu, and Gavin Macriddle. "Edge Intelligence for Data Handling and Predictive Maintenance in IoT." IEEE Access 9: 49355-49371. 2021.

[28] Wang, Xiaodong, Sahil Garg, Hui Lin, Jia Hu, Georges Kaddoum, Md Jalil Piran, and M. Shamim Hossain. "Towards accurate anomaly detection in industrial internet-of-things using hierarchical federated learning." IEEE Internet of Things Journal. 2021.

[29] Li, Jianhua, Lingjian Lyu, Ximeng Liu, Xuyun Zhang, and Xixiang Lv. "FLEAM: A federated learning empowered architecture to mitigate DDoS in industrial IoT." IEEE Transactions on Industrial Informatics. 2021.

[30] Scherz, Paul, and Simon Monk. "Practical electronics for inventors," McGraw-Hill Education, 2016.

MARIA SAMEEN received her BS in Computer Science from the Institute of Business Administration (IBA), Karachi, Pakistan, in 2018, and MS degree in Computer Engineering from Gachon University, South Korea, in 2021. She is a researcher in the Information Security and Machine Learning Lab, Gachon University, South Korea. Her research interests include cybersecurity, network and system security, malware analysis, machine learning, artificial intelligence, and Internet of Things security.

SEONG OUN HWANG received the B.S. degree in mathematics from Seoul National University, in 1993, the M.S. degree in information and communications engineering from the Pohang University of Science and Technology, in 1998, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, in 2004, South Korea. He worked as a Software Engineer with LG-CNS Systems, Inc., from 1994 to 1996. He worked as a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), from 1998 to 2007. He worked as a Professor with the Department of Software and Communications Engineering, Hongik University, from 2008 to 2019. He is currently a Professor with the Department of Computer Engineering, Gachon University. He is an Editor of ETRI Journal. His research interests include cryptography, cybersecurity, and artificial intelligence.