Cluster Computing
https://doi.org/10.1007/s10586-022-03949-w

Robust Federated Learning for execution time-based device model identification under label-flipping attack

Pedro Miguel Sánchez Sánchez[1] · Alberto Huertas Celdrán[2] · José Rafael Buendía Rubio[1] · Gérôme Bovet[3] · Gregorio Martínez Pérez[1]

Received: 13 December 2022 / Revised: 13 December 2022 / Accepted: 19 December 2022

Abstract
The computing device deployment explosion experienced in recent years, motivated by the advances of technologies such as Internet-of-Things (IoT) and 5G, has led to a global scenario with increasing cybersecurity risks and threats. Among them, device spoofing and impersonation cyberattacks stand out due to their impact and, usually, low complexity required to be launched. To solve this issue, several solutions have emerged to identify device models and types based on the combination of behavioral fingerprinting and Machine/Deep Learning (ML/DL) techniques. However, these solutions are not appropriate for scenarios where data privacy and protection are a must, as they require data centralization for processing. In this context, newer approaches such as Federated Learning (FL) have not been fully explored yet, especially when malicious clients are present in the scenario setup. The present work analyzes and compares the device model identification performance of a centralized DL model with an FL one while using execution time-based events. For experimental purposes, a dataset containing execution-time features of 55 Raspberry Pis belonging to four different models has been collected and published. Using this dataset, the proposed solution achieved 0.9999 accuracy in both setups, centralized and federated, showing no performance decrease while preserving data privacy. Later, the impact of a label-flipping attack during the federated model training is evaluated using several aggregation mechanisms as countermeasures. Zeno and coordinate-wise median aggregation show the best performance, although their performance greatly degrades when the percentage of fully malicious clients (all training samples poisoned) grows over 50%.

Keywords Model identification · Federated Learning · Adversarial attacks · Secure aggregation · Insider threat

1 Introduction

Currently, a vast number of devices are deployed worldwide, from smart cars, traffic lights, and security systems, to smart homes and industries. The IoT market has grown to a total of 31 billion connected devices by 2020, with a forecast of $\approx$ 30 billion devices connected to each other by 2023, according to Cisco [1]. One of the main reasons for this growth is the fourth industrial revolution, or Industry 4.0, with the explosion of a set of technologies and paradigms such as 5G, machine and deep learning (ML/DL), robotics, and cloud computing.

The emergence of such technologies poses new challenges to be solved to ensure a safe and efficient environment [2]. In this sense, there are billions of connected devices, many of them performing critical tasks where failures can be fatal, such as autonomous car driving or industrial operations. In addition, the growing popularity of these technologies makes them a desirable target for cybercriminals. Between the possible security threats affecting resource-constrained devices, device impersonation is one of the most serious problems of large organizations with proprietary hardware where one device model could be impersonated for malicious purposes, such as

✉ Pedro Miguel Sánchez Sánchez
pedromiguel.sanchez@um.es

[1] Department of Information and Communications Engineering, University of Murcia, 30100 Murcia, Spain

[2] Communication Systems Group (CSG), Department of Informatics (IfI), University of Zurich UZH, 8050 Zürich, Switzerland

[3] Cyber-Defence Campus within armasuisse Science & Technology, 3602 Thun, Switzerland

Published online: 02 January 2023

industrial espionage. In addition, there are a multitude of counterfeit devices on the market, some of which are difficult to differentiate from the original [3].

To solve these issues, device model and type identification based on performance fingerprinting arises as a solution [4]. The main benefit of device model identification is to prevent third-party attacks such as spoofing, as well as to identify malicious or counterfeit devices. Although there are numerous works in the literature exploring the identification of models from different performance characteristics, such as execution time, network connections or system logs, and leveraging ML/DL for data processing, these solutions mostly require data centralization, making them not suitable for scenarios where data leakage protection and privacy is critical. In this sense, Federated Learning (FL) based techniques have recently gained enormous prominence [5]. In FL approaches, the training data of the ML/DL models remain private and while the locally trained models are shared. Later, these models are aggregated (usually by a central party) into a joint model that goes back to the clients for further training, cyclically repeating the process. This approach improves the privacy of the data, as it does not leave the client, and the communication overhead, as sharing only model parameters is usually less resource-consuming than sharing the complete data used for training.

In addition, there are few datasets modeling the performance of IoT devices for identification [4], and any of them is focused on execution time performance or FL-based scenarios. Moreover, most of the current solutions in the literature do not explore the impact of possible adversarial attacks targeting the ML/DL models during their generation and deployment [6]. These attacks may happen when one of the clients participating in the federation maliciously sends corrupted model updates. These problems have additional importance in FL setups, where the control of the clients is no longer under the entity generating the joint ML/DL model.

Therefore, this work explores the following three main areas to improve the completeness of the literature: (i) the identification of device models using centralized Machine Learning (ML) algorithms and execution time data, (ii) the decentralization of this training using the FL techniques, and (iii) the use of the Adversarial Machine Learning (AML) techniques to evaluate and improve the robustness of the generated models. In this sense, its main contributions are:

- An execution time-based performance dataset collected in 55 different Raspberry Pi (RPi) devices from four different models, and intended for model identification. This dataset is generated using physical devices under normal functioning, reflecting a real scenario where many devices are operating.
- The comparison between a centralized and a federated Multi-Layer Perceptron (MLP) model with identical configuration, only changing its training approach. It is shown how the federated setup maintains an almost identical model identification accuracy of 0.9999, without losing performance and improving data leakage protection and privacy.
- The comparison of different aggregation methods as countermeasure for the federated model under a label-flipping attack. Federated averaging, coordinate-wise median, Krum, and Zeno aggregation methods are compared, showing median and Zeno the best results regarding attack resilience.

The remainder of this paper is structured as follows. Section 2 describes the closest works in the literature, motivating this research. Section 3 explains the procedure followed to extract the model identification data. Later, Sect. 4 compares the performance of a DL-based classifier when it is trained from a centralized and a federated approach. Section 5 explains the adversarial setup followed to test the solution resilience against attacks. Finally, Sect. 6 draws the conclusions extracted from the present research and future lines to explore.

2 Related work

This section will review how the device identification problem has been addressed to date from different approaches and techniques. Likewise, some works in the literature on FL and Adversarial Machine Learning will be analyzed.

Device type and model identification has been widely explored in the literature, with varied data sources and ML/DL-based processing techniques [4]. As one of the closest works to the present one, the authors of [7] proposed a novel challenge-response fingerprinting framework called STOP-AND-FRISK (S & F) to identify classes of Cyber-Physical Systems (CPS) devices and complement traditional CPS security mechanisms based on hardware and OS/kernel. It is exposed that unauthorized and spoofed devices may include manipulated pieces of software or hardware components that may adversely affect CPS operations or collect vital CPS metrics from the network. Another interesting paper showing a fingerprinting technique using hardware performance is [8]. Such a technique is based on the execution times of instruction sequences available in API functions. Due to its simplicity, this method can also be performed remotely. Additionally, the network is the main data source employed in the literature

for device model and type identification [9], as it can be collected from an external gateway.

Regarding the application of FL in device identification, the authors of [10] leveraged FL for device type identification using network-based features. Here, the authors experienced a slightly reduced performance compared to a centralized setup, 0.851 F1-score in the centralized and 0.849 in the federated, but the training process was faster and safer. Additionally, in [11], the authors performed application-type classification based on network traffic using FL to build the models. Although the authors of [12] proposed a distributed solution for network-based model identification, data is shared with an aggregator that performs clustering for model inference. Therefore, no privacy is preserved in this solution.

Moreover, datasets available in the literature for device type or model identification are focused on dimensions such as network connection [13] or radio frequency fingerprinting [14]. However, there is no execution time-based datasets modeling device performance for identification, just some benchmark datasets focused on other tasks [15].

Concerning adversarial ML in FL, the authors of [16] exposed the impossibility of the central server controlling the clients of the federated network. A malicious client could send poisoned model updates to the server in order to worsen learning performance. A new framework for FL is proposed in which the central server learns to detect and remove malicious model updates using a detection model. Finally, the authors of [17] considered the presence of adversaries in their solution for FL-based network attack detection. However, no model identification experiments were carried out.

In conclusion, although each research topic, namely hardware-based model identification, FL, and adversarial ML, has been separately explored. To the best of our knowledge, and as Table 1 shows, there is no work in the literature analyzing device model identification from a federated learning perspective. Besides, there is no dataset focused on model identification based on execution time-based features. Furthermore, there is no solution evaluating the impact of adversarial attacks when some clients are malicious, together with the main aggregation-based attack mitigation techniques.

3 Scenario and dataset creation

This section describes the scenario and the procedure followed to generate the execution time dataset used in the present work. Besides, it provides some insights into the data distribution that can be useful for understanding the model identification performance.

3.1 Scenario description

In total, a setup of 55 RPis from different models but identical software images are employed for data collection, running using *Raspbian 10 (buster) 32 bits* as OS and *Linux kernel 5.4.83*. The generated dataset is composed of 2.750.000 vectors (55 devices $\times$ 50,000 vectors per device). Each vector has two associated labels, one regarding the individual device that generated it and another regarding the model of this device. Data collection was performed under normal device functioning and default frequency and power configuration, where the CPU frequency is automatically adjusted according to the workload. The list of devices contained in the dataset is shown in Table 2.

Table 1 Comparison of the most relevant model identification literature works

Work	Model identification	FL	Adv. attack	Conclusions
[7]	✔ (Hardware-based)	✗	✗	0.9873 average accuracy using correlation-based algorithms to recognize 11 device classes
[8]	✔ (Hardware-based)	✗	✗	+200 computers individually identified based on execution-time statistical comparison
[10]	✔ (Network-based)	✔	✗	0.882 accuracy using a federated LSTM network to identify 10 IoT device types
[11]	– (app identification)	✔	✗	0.92 accuracy using a federated CNN to identify user-level applications
[12]	✔ (Network-based)	✗ (Distributed)	✗	≈0.97 accuracy for clustering-based IoT device type classification
This work	✔ (Hardware-based)	✔	✔ (Label-flipping)	0.9999 accuracy identifying RPi models and adversarial impact analysis.

Table 2 Devices employed in data collection

No. of devices	Model	No. of samples
12	RPi 4 Model B	660,000
22	RPi 3 Model B+	1,210,000
5	RPi 2 Model B	275,000
16	RPi 1 Model B	880,000

3.2 Dataset creation

The generated dataset has been made publicly available [18] for download and research of other authors. The published data includes both identifiers for the RPi model and for individual devices, so new research could be done regarding individual device identification.

For the device performance dataset generation, the CPU performance of the device was leveraged as the data source. In this sense, the time to execute a software-based random number generation function was measured in microseconds.

To minimize the impact of noise and other processes running in the device, the monitored function was executed in groups of 1000 runs a total number of 50,000 times per group. Then, for each 1000-run group, a set of statistical features was calculated, generating a performance fingerprint composed of 50,000 vectors per device. In total, 13 statistical features are calculated: maximum, minimum, mean, median, standard deviation, mode sum, minimum decrease, maximum decrease, decrease summation, minimum increase, maximum increase, and increase summation. Decrease and increase values are calculated as the negative or positive difference between two consecutive values in each 1000-run group. Besides, the device model is added as label. Table 3 shows an example of a vector in the dataset belonging to a Raspberry Pi 4 device.

3.3 Data exploration

Figure 1 shows the data distribution for min, max, mean and median features. It can be observed how the values vary according to the model that generated the vector, resulting in a presumably good model identification performance.

4 Centralized vs federated model identification performance

This section seeks to evaluate firstly the performance of the generated dataset when identifying the different device models in a DL-based centralized setup, and secondly the performance variation when the model is generated in a distributed manner, following an FL-based approach.

4.1 Centralized setup

For the centralized experiment, the dataset described in Section 3 is divided into 80% for training/validation and 20% for testing, without data shuffling. *Min-max normalization* is applied then using the training data to set the boundaries.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

To measure the centralized classification performance, a (*MLP*) classifier is implemented. After several iterations testing different numbers of layers and neurons per layer, the chosen MLP architecture is composed of 13 neurons in the input layer (one per feature), two hidden layers with 100 neurons each one using *relu (Rectified Linear Unit)* as activation function [19], and 4 neurons in the output *softmax* layer (one per model class). *Adam* [20] was used as optimizer with a 0.001 learning rate, and 0.9 and 0.999 as first and second-order moments. Table 4 shows the details of the model.

With this setup, the MLP is trained for 100 epochs using *early stopping* if no validation accuracy improvement occurs in 20 epochs.

Figure 2 shows the confusion matrix resultant of the evaluation of the test dataset. As it can be seen, almost a perfect identification is achieved, with only 15 samples being misclassified out of $\approx$ 550,000 (0.999972 accuracy). These results are aligned with the expectations, as having different CPUs in each RPi model makes the execution time of the same functions different between them. However, model identification performance is not the main focus of the present work, where the priority is to prove the effectiveness of a federated setup and the impact of adversarial attacks and countermeasures.

Table 3 Vector example for a RPi4 device

Min	Max	Mean	Median	SD	Mode	Sum	Min decr.	Max decr.	Decr. sum	Min incr.	Max Incr.	Incr. sum	Model
2.1	12.7	4.2	4.5	1.3	3.5	4221.8	− 113.74	− 8.7	− 0.001	117.8	0.001	7.81	RPi4

Values show the microseconds required to execute a function

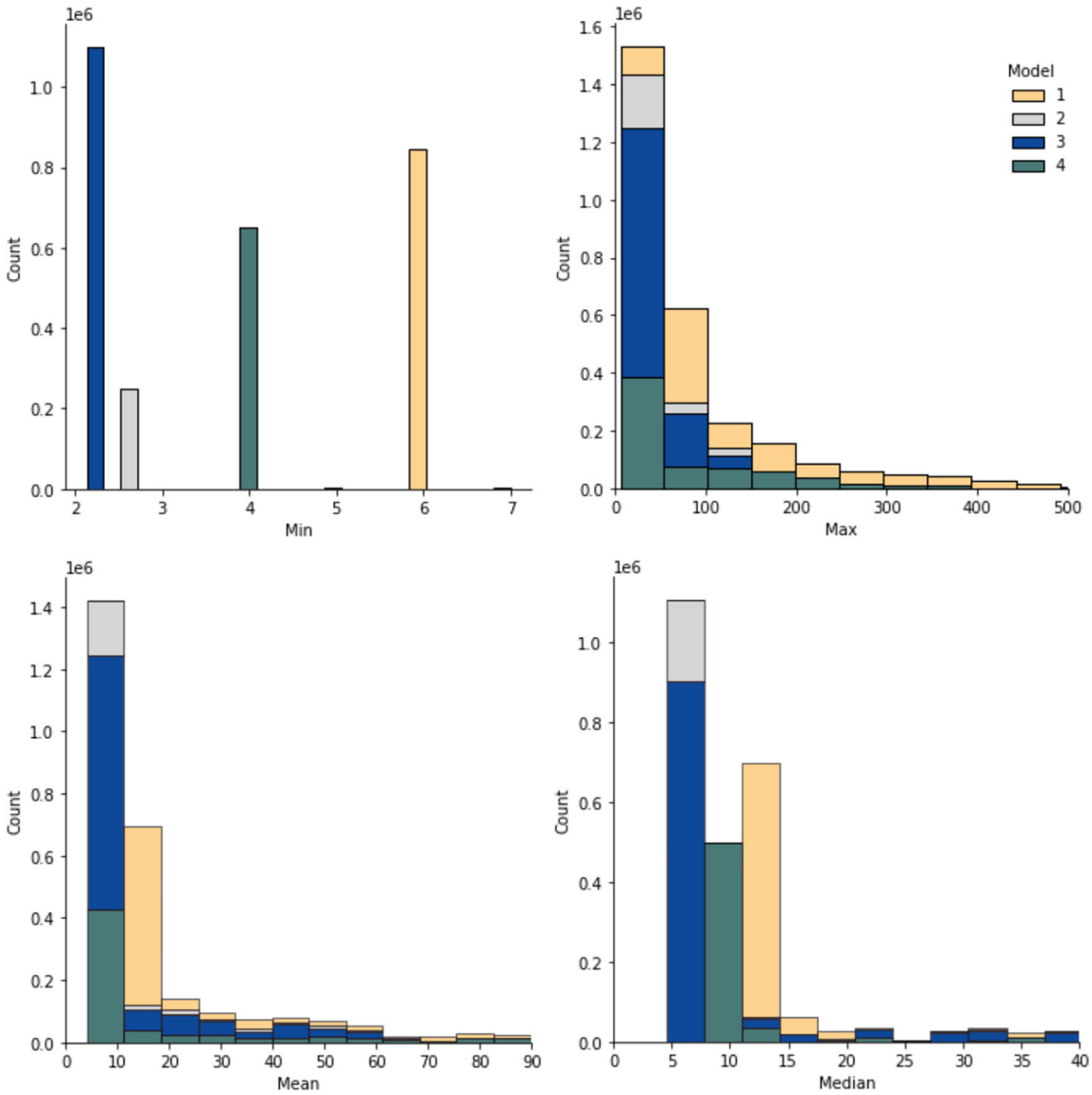


Fig. 1 Min, max, mean and median feature distributions

Table 4 MLP architecture for model identification

Layer	Neurons	Activation
1	13	–
2	100	relu
3	100	relu
4	4	softmax
Optimizer	Adam	

4.2 Federated scenario and results

Once the centralized model has been obtained, the decentralized model is implemented using FL to compare the performance of both approaches. The FL approach is based on horizontal FL, where the clients have datasets with the same features but from different data samples.

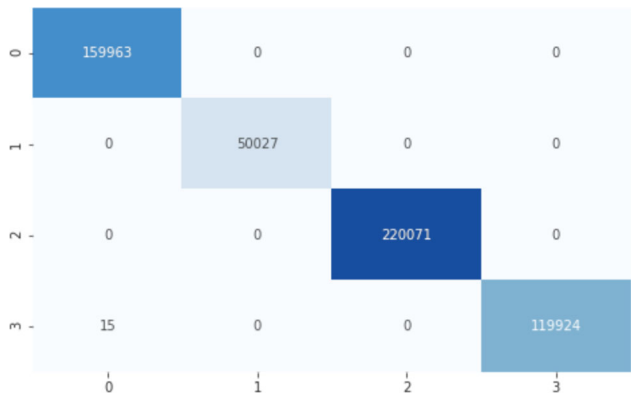


Fig. 2 Centralized evaluation confusion matrix

For implementation, the IBM Federated Learning library [21] is used, which incorporates the necessary tools to perform the training in a decentralized manner.

4.2.1 Scenario

For the decentralization of the training phase, a scenario has been created in which there are 5 independent organizations in which the available data are distributed. Each of them has a certain number of devices belonging to different models, but not all of them have information on all models, i.e. there are organizations that only have devices of type 4 model, others that only have devices of type 2 and 3 models, etc. Figure 3 provides the details of the device distribution in each organization. Therefore, the 5 organizations intend to generate a global model capable of identifying all the existing device models among all of them. This setup leads to a scenario of Non-IID (Non-Independent and Identically Distributed) data, harder to solve with FL as model aggregation will be negatively influenced in the aggregated models are very different from each other.

4.2.2 Federated architecture design

In order to test the performance of an FL-based setup, first it is necessary to define the architecture to be implemented. In this sense, Fig. 4 shows the organization of the different clients which will hold the data and upload their local models to the aggregator in order to cyclically build a common model capable of making predictions based on the local data of all clients.

4.2.3 Performance evaluation

In order to fairly compare the models, the MLP architecture to be trained will be the same as the one used in the centralized model (see Table 4, i.e. the layers will have 13,

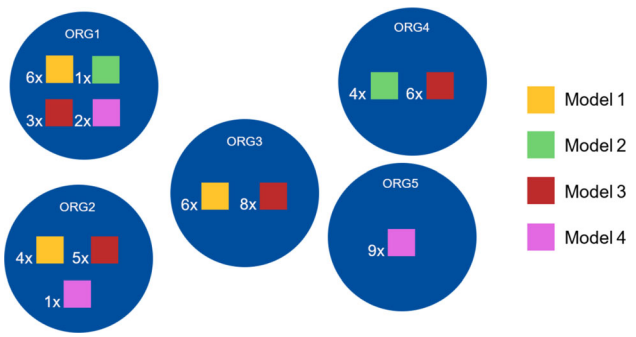


Fig. 3 Data division in the FL scenario

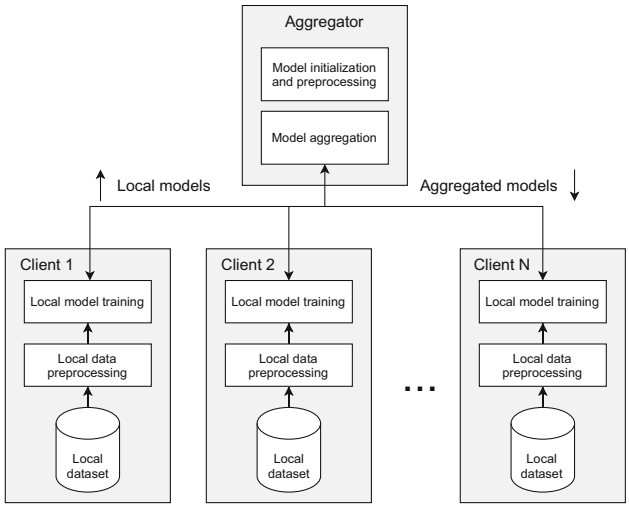


Fig. 4 Designed federated architecture for experimentation

100, 100, 100, and 4 neurons, from input to output. As aggregation method, Federated Averaging is applied as proposed in [22]. As initialization step, the aggregation server performs two tasks: (1) to initialize the weights of the model that the clients will start to train, so all clients start from the same setup; (2) to retrieve for each client its min-max values of each feature for common dataset normalization, having a *min-max normalization* for each dataset x in organization o $\in x$ defined as:

$$x'_o = \frac{x_o - min(x_{i\in[n]})}{max(x_{i\in[n]}) - min(x_{i\in[n]})} \tag{2}$$

Algorithm 1 defines the iterative training process for the model generation, assuming previous dataset normalization. Each client performs local updates of the model and

Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate; w are the model weights; P_k is the local dataset of client k. [22]

Server executes:
  initialize w_0
  for each round t = 1,2... **do**
    $m \leftarrow max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of m clients)
    for each client $k \in S_t$ **in parallel do**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$ //Aggregation

ClientUpdate(k, w): //*Run on client k*
  $B \leftarrow$ (split P_k into batches of size B)
  for each local epoch i from 1 to W **do**
    for batch $b \in B$ **do**
      $w \leftarrow w - \eta \bigtriangledown l(w;\ b)$ //Local update
  return w to server

returns them to the server for aggregation, repeating then the process for the desired number of rounds.

The training process was executed for 90 federated rounds, with one epoch per round. Figure 5 shows the evolution of the local validation accuracy for each one of the clients during the training process. It can be appreciated how the maximum performance is reached around epoch 50, and then the accuracy scores for each client keep oscillating between 0.95 and 1 until round 90.

Regarding performance, Fig. 6 shows the results of the test dataset evaluation, the same dataset than in the centralized setup. Here, the results are almost identical, with only 17 errors in $\approx$ 55,0000 test samples and an accuracy of 0.999969.

From the previous results, the main conclusion can be extracted: no performance loss has been introduced in the resultant model due to the application of an FL-based approach. Besides, as no data has left each organization in the process, the privacy of the information has been kept private successfully.

5 Adversarial attack and robust aggregation

After testing the effectiveness of FL, its robustness will be tested using adversarial attacks, specifically the label-flipping technique, using different aggregation algorithms in order to see which one best fits the proposed scenario in the presence of attacks.

5.1 Label flipping attack

The label-flipping adversarial technique is applied during the training process, using the same scenario described above with the difference that this time part of the data will be poisoned.

In this sense, the federated training is carried out by poisoning 25, 50, 75, and 100% of the data of 1, 2, and 3

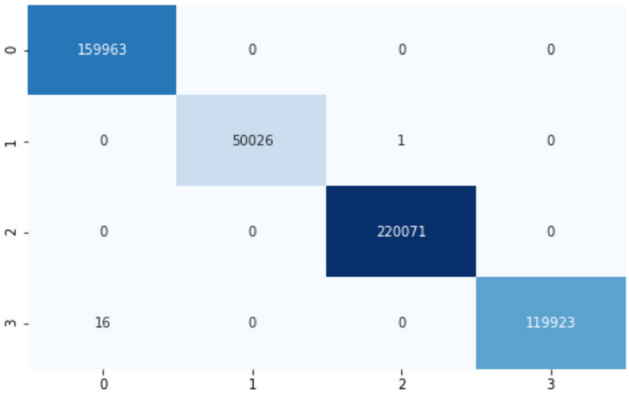


Fig. 6 Federated evaluation confusion matrix

different organizations, representing 20%, 40%, and 60% malicious clients, respectively. These configurations are used because potential malicious clients may not poison all their data and just one portion, in order to go undetected and make their activity more difficult to identify. So, a total of 12 adversarial scenarios have been created (4 poising percentages $\times$ 3 possible malicious organizations). This setup is generated by modifying the labels of the training data, changing the value of each label to a random value between 1 and 4 which is not the value of the original label. The poisoned organizations are ORG1, ORG2, and ORG4 (in that order for 1, 2, and 3 malicious clients).

Figure 7 shows the results when FedAvg is applied as the aggregation algorithm in the 12 previous adversarial scenarios (as well as when no label-flipping attack is applied).

As can be seen, aggregation by averaging offers good performance up to 50% poisoning, maintaining an accuracy over 0.9. However, accuracy drops rapidly to hit rates close to 0% when the poisoning is 75% or higher. Therefore, FedAvg cannot be considered a robust aggregation method in the presence of a label-flipping attack. Next, 3 different aggregation methods will be analyzed in the following in order to check which one offers better performance.

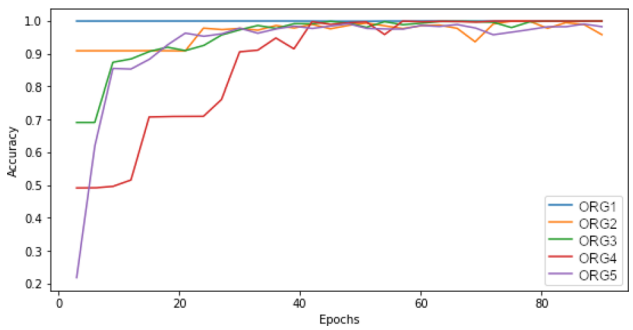


Fig. 5 Federated training validation accuracy evolution

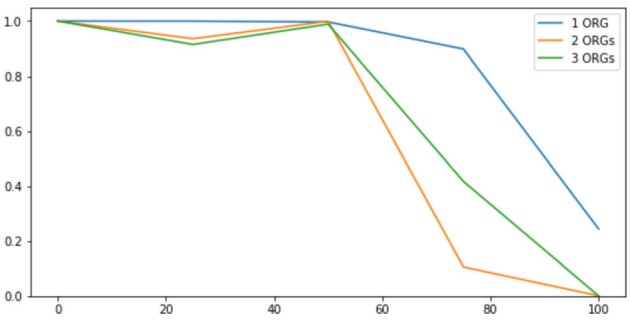


Fig. 7 FedAvg performance against label-flipping attack (X axis depicts the poisoning percentage, Y axis depicts accuracy)

5.2 Robust aggregation methods

Next, several aggregation methods focused on improving the model resilience to malicious clients will be evaluated and compared to the default FedAvg algorithm.

5.2.1 Coordinate-wise median aggregation

Coordinate-wise median [23] follows the scheme of the aggregation by average with the difference that the combination of the weights is done by calculating the median of each weight of the local models. In short, following Algorithm 1, the averaging aggregation step is substituted by a median operation.

Next, Fig. 8 depicts the accuracy results when the different attack setups are applied when using median aggregation. Coordinate-wise median follows a similar pattern to FedAvg aggregation, dropping from 50% poisoning rate. However, it has performed better especially when there is only one poisoned organization (20% malicious clients). While FedAvg dropped to 0.20-0.40, the median has remained around $\approx$ 0.9.

5.2.2 Krum aggregation

The idea behind Krum [24] is to select as the global model the local model which is most similar to the rest. The idea is that even if the selected model is a poisoned model, the impact would not be so great since it would be similar to other models that are probably not poisoned. The aggregator calculates the sum of the distances between each model and its closest local models. Krum selects the local model with the smallest sum of distances as the global model. Figure 9 shows the results when Krum is applied as the aggregation algorithm.

As can be seen, Krum has remained constant for all configurations with an accuracy of 0.6896. This is because this aggregation method chooses a single local model as the global model and discards the information from the rest of the local models. Therefore, what is happening is that it always chooses the same local model, and this one belongs to an organization that has not been poisoned, so the hit rate remains constant. Figure 10 shows that the resulting global model only recognizes device models of types 0 and 2.

On the other hand, this organization has not been poisoned, which explains that the performance remains constant since the resulting global model is identical regardless of the percentage of poisoning. Therefore, it can be concluded that Krum is selecting the resulting local model of organization 3 in all scenarios, losing the information regarding the classes not seen in this organization (see Fig. 3).

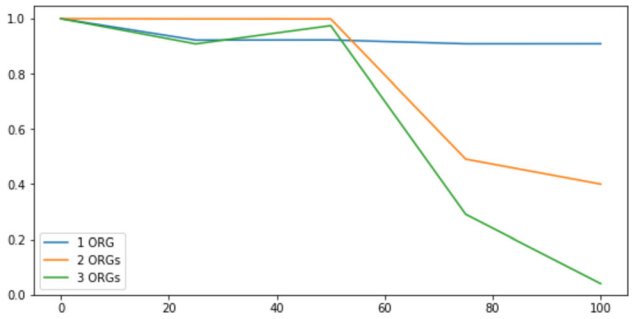


Fig. 8 Coordinate-wise median aggregation performance against label-flipping attack (X axis depicts the poisoning percentage, Y axis depicts accuracy)

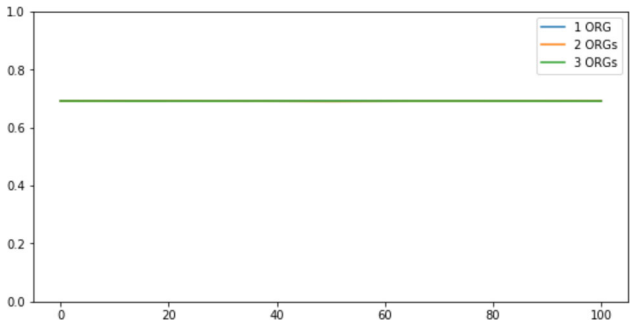


Fig. 9 Krum performance against label-flipping attack (X axis depicts the poisoning percentage, Y axis depicts accuracy)

5.2.3 Zeno aggregation

Zeno [25] is suspicious of potentially malicious organizations and uses a ranking-based preference mechanism. The number of malicious organizations can be arbitrarily large, and only the assumption that 'clean' organizations exist (at least one) is used. Each organization is ranked based on the estimated descent of the loss function. The algorithm then aggregates the organizations with the highest scores. The score roughly indicates the reliability of each organization. In this sense, it could be seen as a combination of Krum and averaging aggregation mechanisms. Figure 11 shows

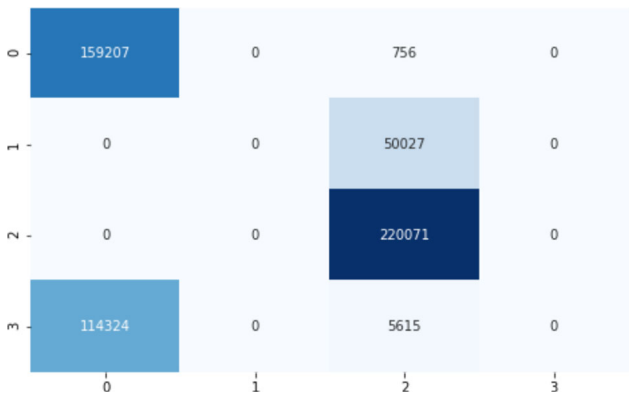


Fig. 10 Krum confusion matrix during the label-flipping attack

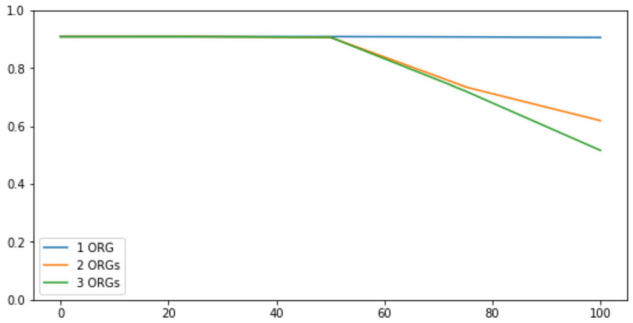


Fig. 11 Zeno performance against label-flipping attack (X axis depicts the poisoning percentage, Y axis depicts accuracy)

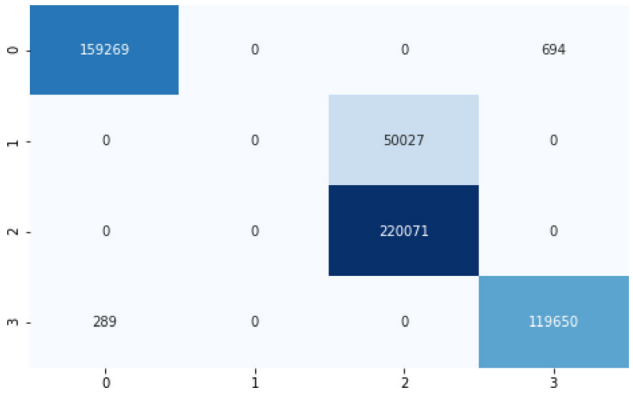
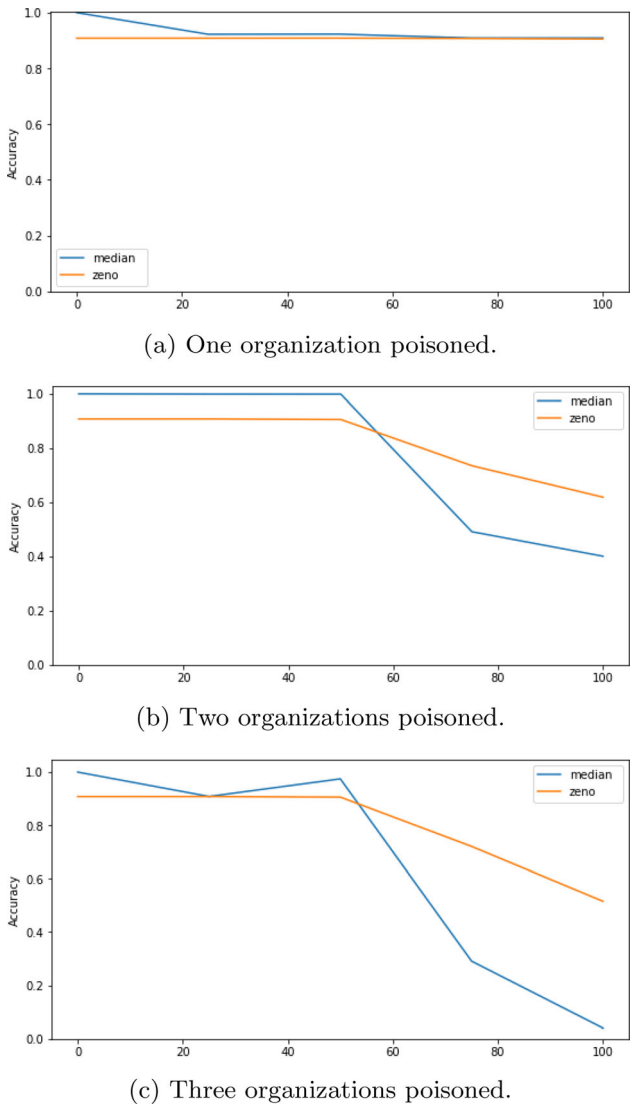


Fig. 12 Zeno confusion matrix with one malicious client

the results when Zeno is applied as the aggregation algorithm.

In this case, Zeno has outperformed the aggregation by median and Krum when only one client is malicious (20% of the total), achieving 0.9072 accuracy. Zeno remains constant without being altered by this attack when there is only one poisoned organization. Figure 12 shows the confusion matrix of Zeno when one client is malicious. It can be appreciated how the performance decrease comes from the impossibility of classifying the second class, the one underrepresented in the scenario, as there are only 5 RPi2 in the dataset.

When there are 2 or 3 poisoned organizations, Zeno performance drops once the poisoning rate reaches 75% and 100%. However, it still manages to maintain an acceptable performance above 0.50, considering the degree of the attack. Figure 13 compares the performance evolution of Zeno and coordinate-wise median with different number of poisoned organizations. As it can be appreciated, median performance is higher in all scenarios until the poisoning percentage goes above 50%. After that, Zeno shows a better or equal performance in all cases, being the greatest difference when three organizations are completely malicious (60% malicious clients).



(a) One organization poisoned.

(b) Two organizations poisoned.

(c) Three organizations poisoned.

Fig. 13 Zeno and coordinate-wise median aggregation with different poisoned clients (X axis depicts poisoning percentage, Y axis depicts accuracy)

6 Conclusions and future work

In the present work, it has been demonstrated that it is possible to identify device models using only statistical data concerning the CPU execution time of the device. An MLP model has been obtained capable of identifying four RPi device models with a 99.99% accuracy rate. Besides, the effectiveness of the FL technique has been tested against centralized learning. For this setup, a scenario has been proposed where a total of five organizations aim to create a model capable of identifying the device models without sharing the actual data with each other. The resulting model has obtained identical performance in both cases, centralized and distributed. Thus taking advantage of the benefits offered by FL, training data privacy and data

security preserving model while maintaining the efficiency of the model obtained through a traditional approach. On the other hand, different aggregation algorithms have been tested in order to check which one best fits the proposed scenario facing a label-flipping attack. Zeno has turned out to be the best-performing aggregation method in the presence of attacks due to combining the Krum and mean aggregation methods. By selecting the m best models and aggregating them using mean aggregation, less information is lost than with Krum by ignoring certain organizations that are considered malicious. Finally, the data collected for the previous experimentation has been made publicly available due to the lack of performance fingerprinting datasets focused on device identification and prepared for FL-based setups.

In future work, the efforts will be focused on experimentation with more types of device models with more complex scenarios, such as making each device a single client instead of being grouped into organizations. Regarding device identification, it is planned to focus on identifying individual devices with a high hit rate and not just identifying device models, as well as testing other modes of identification by collecting data from hardware elements other than the CPU. It would also be interesting to poison the local model weights instead of the local data (model poisoning) or experiment with other adversarial attack techniques, such as Evasion attacks, where the goal is to trick the model once it is trained and not to poison the training process.

Acknowledgements This work has been partially supported by (a) the Swiss Federal Office for Defense Procurement (armasuisse) with the TREASURE and CyberSpec (CYD-C-2020003) projects and (b) the University of Zürich UZH.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data availability The datasets generated and analyzed during the current study are available in the RPi model device identification Mendeley Data repository, https://doi.org/10.17632/vr9wztmfxg.2.

References

1. Cisco: Cisco Annual Internet Report (2018–2023) White Paper (2020). https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html. Accessed 22 Nov 2021
2. Nižetić, S., Šolić, P., González-de, D.L.D.-I., Patrono, L., et al.: Internet of Things (IoT): opportunities, issues and challenges towards a smart and sustainable future. J. Clean. Prod. **274**, 122877 (2020)
3. Negka, L., Gketsios, G., Anagnostopoulos, N.A., Spathoulas, G., Kakarountas, A., Katzenbeisser, S.: Employing blockchain and physical unclonable functions for counterfeit IoT devices detection. In: Proceedings of the International Conference on Omni-Layer Intelligent Systems, pp. 172–178 (2019)
4. Sánchez Sánchez, P.M., Jorquera Valero, J.M., Huertas Celdrán, A., Bovet, G., Gil Pérez, M., Martínez Pérez, G.: A survey on device behavior fingerprinting: data sources, techniques, application scenarios, and datasets. IEEE Commun. Surv. Tutorials **23**(2), 1048–1077 (2021). https://doi.org/10.1109/COMST.2021.3064259
5. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. **10**(2), 1–19 (2019)
6. Nguyen, T.D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., Sadeghi, A.: DÏot a federated self-learning anomaly detection system for IoT. In: 39th IEEE International Conference on Distributed Computing Systems, pp. 756–767 (2019). https://doi.org/10.1109/ICDCS.2019.00080
7. Babun, L., Aksu, H., Uluagac, A.S.: Cps device-class identification via behavioral fingerprinting: from theory to practice. IEEE Trans. Inf. Forensics Security **16**, 2413–2428 (2021)
8. Sanchez-Rola, I., Santos, I., Balzarotti, D.: Clock around the clock: time-based device fingerprinting. In: 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 1502–1514 (2018)
9. Meidan, Y., Bohadana, M., Shabtai, A., Guarnizo, J.D., Ochoa, M., Tippenhauer, N.O., Elovici, Y.: Profiliot: A machine learning approach for IoT device identification based on network traffic analysis. In: Proceedings of the Symposium on Applied Computing, pp. 506–509 (2017)
10. He, Z., Yin, J., Wang, Y., Gui, G., Adebisi, B., Ohtsuki, T., Gacanin, H., Sari, H.: Edge device identification based on federated learning and network traffic feature engineering. IEEE Trans. Cogn. Commun. Netw. **8**(4), 1898–1909 (2021)
11. Mun, H., Lee, Y.: Internet traffic classification with federated learning. Electronics **10**(1), 27 (2021)
12. Thangavelu, V., Divakaran, D.M., Sairam, R., Bhunia, S.S., Gurusamy, M.: DEFT: a distributed IoT fingerprinting technique. IEEE Internet Things J. **6**(1), 940–952 (2019)
13. Aksoy, A., Gunes, M.H.: Automated IoT device identification using network traffic. In: ICC 2019-2019 IEEE International Conference on Communications (ICC), pp. 1–7. IEEE, Piscataway (2019)
14. Abbas, S., Nasir, Q., Nouichi, D., Abdelsalam, M., Talib, M.A., Waraga, O.A., et al.: Improving security of the Internet of Things via RF fingerprinting based device identification system. Neural Comput. Appl. **33**, 14753–14769 (2021)
15. Varghese, B., Wang, N., Bermbach, D., Hong, C.-H., Lara, E.D., Shi, W., Stewart, C.: A survey on edge performance benchmarking. ACM Comput. Surv. **54**(3), 66:1-66:33 (2021). https://doi.org/10.1145/3444692
16. Li, S., Cheng, Y., Wang, W., Liu, Y., Chen, T.: Learning to detect malicious clients for robust federated learning. arXiv preprint (2020). arXiv:2002.00211 (2020)

17. Rey, V., Sánchez, P.M.S., Celdrán, A.H., Bovet, G., Jaggi, M.: Federated learning for malware detection in IoT devices. arXiv preprint (2021). arXiv:2104.09994
18. Bovet, G., Sánchez Sánchez, P.M.: RPi model device identification (2021). https://doi.org/10.17632/vr9wztmfxg.2
19. Agarap, A.F.: Deep learning using rectified linear units (RELU). arXiv preprint (2018).arXiv:1803.08375
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint (2014). arXiv:1412.6980
21. Ludwig, H., Baracaldo, N., Thomas, G., Zhou, Y., Anwar, A., Rajamoni, S., Ong, Y., Radhakrishnan, J., Verma, A., Sinn, M., et al.: Ibm federated learning: an enterprise framework white paper v0. 1. arXiv preprint (2020). arXiv:2007.10987
22. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication—efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, FL (2017)
23. Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-robust distributed learning: towards optimal statistical rates. In: International Conference on Machine Learning, PMLR, pp. 5650–5659 (2018)
24. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 118–128 (2017)
25. Xie, C., Koyejo, S., Gupta, I.: Zeno: distributed stochastic gradient descent with suspicion-based fault-tolerance. In: International Conference on Machine Learning, PMLR, pp. 6893–6901 (2019)

Pedro Miguel Sánchez Sánchez is pursuing his Ph.D. in computer science at the University of Murcia. He received the MSc degree in Computer Science from the University of Murcia, Spain. His research interests focus on continuous authentication, networks, 5G, cybersecurity, and machine learning and deep learning.

Alberto Huertas Celdrán is senior researcher at the Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH. He received the M.Sc. and Ph.D. degrees in Computer Science from the University of Murcia, Spain. His scientific interests include cybersecurity, machine and deep learning, continuous authentication, and computer networks.

José Rafael Buendía Rubio received the B.Sc. degree in computer science from the University of Murcia. His research interests are focused on device identification, cybersecurity and the application of machine learning and deep learning to the previous fields.

Gérôme Bovet received his Ph.D. in networks and computer systems from Telecom ParisTech, France, in 2015, and an Executive MBA from the University of Fribourg, Switzerland in 2021. He is the head of data science for the Swiss Department of Defense, where he leads a research team and portfolio of about 30 CyberDefence projects. His work focuses on ML and DL approaches, with an emphasis on anomaly detection, adversarial and collaborative learning applied to data gathered by IoT sensors.

Gregorio Martínez Pérez is Full Professor in the Department of Information and Communications Engineering of the University of Murcia, Spain. His scientific activity is mainly devoted to cybersecurity and networking. He is working on different national (14 in the last decade) and European IST research projects (11 in the last decade) related to these topics, being Principal Investigator in most of them. He has published 200+ papers in international conference proceedings, magazines and journals.