Information Systems Frontiers (2025) 27:367–390
https://doi.org/10.1007/s10796-023-10443-0

A Fully Privacy-Preserving Solution for Anomaly Detection in IoT using Federated Learning and Homomorphic Encryption

Marco Arazzi[1] · **Serena Nicolazzo**[2] · **Antonino Nocera**[1]

Accepted: 17 October 2023 / Published online: 14 November 2023

Abstract

Anomaly detection for the Internet of Things (IoT) is a very important topic in the context of cyber-security. Indeed, as the pervasiveness of this technology is increasing, so is the number of threats and attacks targeting smart objects and their interactions. Behavioral fingerprinting has gained attention from researchers in this domain as it represents a novel strategy to model object interactions and assess their correctness and honesty. Still, there exist challenges in terms of the performance of such AI-based solutions. The main reasons can be alleged to scalability, privacy, and limitations on adopted Machine Learning algorithms. Indeed, in classical distributed fingerprinting approaches, an object models the behavior of a target contact by exploiting only the information coming from the direct interaction with it, which represents a very limited view of the target because it does not consider services and messages exchanged with other neighbors. On the other hand, building a *global* model of a target node behavior leveraging the information coming from the interactions with its neighbors, may lead to critical privacy concerns. To face this issue, the strategy proposed in this paper exploits Federated Learning to compute a global behavioral fingerprinting model for a target object, by analyzing its interactions with different peers in the network. Our solution allows the training of such models in a distributed way by relying also on a secure delegation strategy to involve less capable nodes in IoT. Moreover, through homomorphic encryption and Blockchain technology, our approach guarantees the privacy of both the target object and the different workers, as well as the robustness of the strategy in the presence of attacks. All these features lead to a secure fully privacy-preserving solution whose robustness, correctness, and performance are evaluated in this paper using a detailed security analysis and an extensive experimental campaign. Finally, the performance of our model is very satisfactory, as it consistently discriminates between normal and anomalous behaviors across all evaluated test sets, achieving an average accuracy value of 0.85.

Keyword Internet of Things, Federated Learning, Blockchain, Autonomy, Reliability, Machine Learning, Privacy, Homomorphic Encryption

1 Introduction

The massive distribution of smart and interconnected devices is making us spectators and actors, at the same time, of a new world of application scenarios inside the Internet of Things (IoT, hereafter). However, as the pervasiveness and autonomy of smart things grow, cyber attacks are becoming more and more dangerous and complex (Adat et al., 2018), demanding security approaches based on always improved and sophisticated techniques. This crucial aspect has to be tackled because security and privacy concerns act as inhibitors of this market's future expansion and evolution (Al-Sarawi et al., 2020).

A recent solution to make IoT more robust to possible security threats and misuse is the computation of *devices fingerprint*, used to detect the object *anomalies* caused by attacks, hardware deterioration, or malicious software modifications (Sánchez et al., 2021). Previous strategies in this context leveraged features derived from device information (i.e., device name, device type, manufacturer information, serial number, and so forth) and other basic networking data to model the identity of an IoT node (Oser et al., 2018; Kohno et al., 2005). More recent approaches, based on Machine Learning (ML, hereafter) and Deep Learning (DL, hereafter)

These authors contributed equally to this work.

✉ Antonino Nocera
antonino.nocera@unipv.it

Extended author information available on the last page of the article

techniques, aim at modeling a complete profile of a *thing*, composed not only of device and network information but also of the hidden and unique patterns in the behavior that a node reveals when it interacts with other peers. This so called *behavioral fingerprint* is more difficult to be forged by a malicious adversary, increasing the probability of detecting potential misbehavior that may arise due to cyber attacks, system faults, or misconfigurations (Aramini et al., 2022; Bezawada et al., 2018; Ferretti et al., 2021; Celdrán et al., 2022).

Most of the approaches based on behavioral fingerprinting fall into two different groups. The first set is composed of centralized solutions in which a single hub is in charge of training and executing ML algorithms to assess the fingerprint of all the devices of the network. Therefore, due to the use of end-to-end encryption, these solutions cannot take into consideration features obtainable by analyzing private message payloads exchanged between every pair of nodes (Hamad et al., 2019; Miettinen et al., 2017). A second group consists of distributed approaches in which a comprehensive profile can be built, but only concerning a single node point of view (i.e., the ML model is trained and executed by a node, based on its direct interactions with a target node) (Aramini et al., 2022; Ferretti et al., 2021).

To overcome these limitations, in this paper, we face the challenge of designing a global model for behavioral fingerprinting considering the information from multiple nodes without centralizing the solution in a single super-node. To do so, we leverage the novel paradigm of Federated Learning (FL, for short) (Yang et al., 2019). Generally, FL is a distributed collaborative AI approach that allows the training of models through the coordination of multiple devices with a central server, acting as an *aggregator*, without the need to share the actual datasets (Nguyen et al., 2021).

In particular, in an IoT scenario, an aggregator can coordinate multiple objects, called workers, to perform neural network training. The main steps can be summarized as follows. First, the aggregator initializes a shared global model with random parameters and broadcasts it to the worker nodes. Secondly, for several iterations, each worker computes its individual model update, leveraging its local dataset. Once the gradient is computed the aggregator receives all model updates and combines them into an aggregated global model. Finally, this global update will be downloaded by the workers to compute their next local update. The steps above are repeated until the global training is complete.

In our paper, we apply this approach to an IoT scenario in which devices with different computational capabilities can cooperate. In particular, the worker devices, in charge of training local ML models, should be powerful devices with sufficient computational capability, memory, and stability. The role of the aggregators, instead, is distributed among multiple devices that can have high or medium computational capabilities. Observe that, each aggregator collects information from workers to create a global model for one or more targets, but a target node can have only one aggregator. In this way, FL can be simply applied to an IoT environment in the form of a "distributed aggregation" architecture, that involves multiple aggregation servers receiving local learning model updates from their associated devices (Khan et al., 2021).

This approach presents several points of strength. First off, global behavioral fingerprints can be computed for a target node by considering aspects captured and modeled by all its peers. This strategy allows for enhanced learning accuracy rates. Approach scalability is also improved due to the distributed learning nature of FL. Moreover, the raw data are not required for the training on the aggregator side, thus minimizing the leakage of sensitive information to external third parties.

However, the application of this strategy can introduce further privacy concerns arising from the exposure of side-channel information. For instance, all the workers involved in the learning task would expose their interactions with the target, and the aggregator would know the identity of the monitored objects.

In this paper, we try to face this further issue by designing a Secure Multi-party Computation (SMC, for short) scheme based on Homomorphic Encryption (HE, for short) and its properties. Unlike conventional encryption algorithms such as Advanced Encryption Standard (AES) or Rivest-Shamir-Adleman (RSA), HE has been designed to perform operations over encrypted data (Gentry, 2009), proving end-to-end IoT dataflow privacy. In general, HE has been applied to IoT scenarios to securely store data in public clouds, where computations, such as the training and execution of ML algorithms, can be performed without deciphering and accessing the user's data (Kim et al., 2018). In our approach, we make use of HE during a safe starting phase. We assume that this phase has a sufficient duration to gather enough data to train ML models in an environment in which the target node is free from possible attacks. Specifically, the main steps of this stage can be summarized as follows.

Every node with sufficient computation capability to train an ML model contacts the target node (for which it wants to compute the behavioral fingerprint) to exchange a message containing the necessary identifier parameters encrypted with a homomorphic hash function.

After this step, the worker nodes query the Blockchain to discover the identity of the aggregator node for the considered target. In our solution, we leverage Blockchain and smart contracts technology for a number of tasks to make it fully distributed. In particular, Blockchain is exploited to implement a reputation mechanism to: *(i)* monitor aggregator nodes at a global level and *(ii)* store malicious nodes' information resulting from the application of our strategy.

To achieve this goal, our approach leverages a consolidated practice, indeed, Blockchain smart contracts are already being used to control and secure IoT devices (Christidis & Devetsikiotis, 2016; Khan & Salah, 2018), and, in addition, lightweight adaptations of a Blockchain have been designed to support resource-constrained smart things (Corradini et al., 2022). As for the reputation mechanism, although this function is orthogonal to our approach, several proposals can be used to provide forms of trust in an IoT network (Corradini et al., 2022; Dedeoglu et al., 2019; Pietro et al., 2018). Nevertheless, in our solution, we adapt an existing schema by allowing nodes to assign a trust score *(i)* to their peers based on the analysis of their behavior through the proposed behavioral fingerprinting model, and *(ii)* to an aggregator according to its performance during the training phase.

With that said, leveraging information exchanged through a refined use of HE properties, worker nodes can identify a common aggregator and, this last can, then, group together the ones with common learning tasks. In our solution, the steps above are carried out by maintaining private all the side information, as a matter of fact, to realize a fully privacy-preserving solution, neither the aggregator must know the identity of the target node, nor the different workers should know each other. Finally, as stated before, in our heterogeneous IoT environment all these devices, even less powerful ones, can benefit from our approach by delegating several tasks of our schema to more capable devices. In our strategy, also this additional facility must be privacy-preserving.

The outline of this paper is as follows. In Section 2, we illustrate the literature related to our approach. In Section 3, we give a general overview of our reference IoT model and describe the proposed framework in detail. In Section 4, we analyze our security model. In Section 5, we present the set of experiments carried out to test our approach and show its performance. Finally, in Section 6, we discuss the limitations of our paper, draw our conclusions, and present possible future works related to our research efforts. In the following, we list the main challenges faced and describe the insightful contributions provided.

1.1 Challenges and Contribution

As described above, the challenges faced by our proposal and its main contributions are numerous and we can summarize them as follows:

- Dynamic threat landscape. IoT devices are constantly updated and released. Nevertheless, vulnerability exploitation is developed at a similarly high pace. This makes the threats against this context highly dynamic and difficult to foresee. We tackle this issue by proposing a behavioral fingerprinting model able to monitor the hidden and unique patterns of the behavior of a node in a network. This tailored countermeasure appears suitable for a constantly changing attack surface.
- Increase security. We improve the accuracy of behavioral fingerprinting models by building a comprehensive object profile. Indeed, adopting a solution based on FL allows us to evaluate the behavior of an object across different services and leverage the interaction with multiple peers.
- Solution scalability. Scalability is an issue that affects various aspects of behavior monitoring approaches, especially in the context of IoT. We face this problem by adopting a FL strategy aiming at distributing the monitoring tasks across the nodes of the network.
- Lack of interaction data. IoT devices generate traffic by infrequent user interactions. FL strategy empowers nodes with global models generated from the aggregation of different contributions.
- Autonomy. The IoT scenario demands a growing number of tasks carried out without the need for human intervention. We leverage Blockchain and smart contract technology for several steps in our approach to distribute the computation and increase object autonomy.
- Privacy of data. IoT devices exchange sensitive information, hence the privacy aspects related to behavioral data and corresponding models play a key role. We adopt FL to secure data during the training of behavioral fingerprinting models. More importantly, we take a step forward in maintaining the private identity of target nodes and workers leveraging a homomorphic encryption-based strategy.
- IoT device heterogeneity. Many IoT devices have limited capabilities in terms of available memory, computing resources, and energy and, therefore, they are not capable of performing complex algorithms. Through our secure delegation solution also less capable devices can benefit from our approach in a privacy-preserving way.

2 Related Works

With the growing complexity and pervasiveness of IoT-based solutions, the surface and the impact of possible attacks against this scenario are increasing as well (Hassija et al., 2019; Li et al., 2015). In the last years, researchers have studied novel countermeasures to the most disparate type of threats to IoT devices (Buccafurri et al., 2016; Kozlov et al., 2012; Sicari et al., 2016; Tweneboah-Koduah et al., 2017), and the latest ones are involving also Machine Learning and Deep Learning techniques (Al-Garadi et al., 2020; Cauteruccio et al., 2019). In this context, a recent trend is to develop ML and DL algorithms to model peculiar characteristics of target objects to detect compromised devices within a network. The ensemble of these features, that an IoT device possesses and reveals when it interacts with other

objects over a network, represents the so called *fingerprint*. Classical device fingerprinting comprehends soft identities, such as: device name, device type, manufacturer information, serial number, network address, and other features that can be derived from different types of networking information. For instance, the authors of (Oser et al., 2018) identified 19 features that can be used to assess the security level of an object directly from the data-link header of 802.11 messages. Also physical layer information is used, for instance, the work illustrated in (Radhakrishnan et al., 2014) focuses on the analysis of the physical aspects of devices, like inter-arrival times of different packets, to fingerprint them. An evolution of such an approach that cannot be very easily cloned by a malicious adversary, is represented by behavioral fingerprinting (Aramini et al., 2022; Bezawada et al., 2018; Celdrán et al., 2022; Ferretti et al., 2021). This type of technique leverages application-level information to extract features concerning the interaction among the devices and, hence, their networking behavior. In particular, in (Bezawada et al., 2018) the authors leverage a number of features extracted from the network traffic of the device to train an ML model that can be used to detect similar device types. The work presented in (Celdrán et al., 2022) illustrates a detection framework that applies device behavioral fingerprinting and ML to detect anomalies and classify different threats, such as: botnets, rootkits, backdoors, and ransomware affecting real IoT spectrum sensors. As for the work presented in (Aramini et al., 2022), it describes an enhanced behavioral fingerprinting model consisting of a fully decentralized scenario, where it is possible to exploit the features derived from the analysis of packet payloads (for instance, different types of devices and their traffic characteristics) and message content as well. Still, there exist challenges in terms of the performance of ML-based fingerprinting solutions able to detect a forged or corrupted smart thing in the network. The causes are related to scalability, security, and privacy issues and also to the fact that an object can model the behavior of another object concerning its single point of view (i.e., the ML algorithm used is thought to evaluate only the services and messages from the interaction of the two things) (Sánchez et al., 2021).

Hence, a new perspective that can comprehend the whole behavior of an object is demanding. Moreover, classical ML techniques require centralized data collection and processing that may not be feasible in IoT application scenarios due to the high scalability of modern IoT networks, growing data privacy concerns, and heterogeneity of devices. To face these issues and allow a collaborative ML approach, Federated Learning (Khan et al., 2021; Nguyen et al., 2021; Yang et al., 2019) solutions have emerged with the aim of distributing ML algorithm execution without the need for data sharing. For instance, (Rey et al., 2022) shows a framework that uses FL to detect malware affecting IoT devices using multi-layer perceptron and autoencoder neural network architectures. Whereas the authors of (Preuveneers et al., 2018) studied FL to design an intrusion detection system. This work also includes Blockchain technology to mitigate the problems faced in adversarial FL, however it does not focus specifically on IoT devices. Also the authors of (Nguyen et al., 2019) used FL, their aim is to build a distributed system for detecting compromised IoT devices through an anomaly detection-based approach. It consists of a simple fingerprint of the device based on network packets able to monitor changes caused by network attacks. All the above works exploit FL for a different goal concerning ours. To the best of our knowledge, no previous works have used FL for behavioral fingerprinting computation.

Till now we described how the problem of scalability and performances of behavioral fingerprinting computation can be faced through FL. But other challenges arise in this new IoT scenario, for instance, the privacy of data exchanged by things.

To face the risk of privacy leakage of sensitive information in the IoT caused by the centralized servers' architecture and the weakness and heterogeneity of devices and security protocols, researchers have begun to exploit the potentiality of Homomorphic Encryption (Peralta et al., 2019; Shrestha & Kim, 2019). For instance, the work presented in (Peralta et al., 2019) shows a possible application of HE to perform computations in the cloud maintaining data privacy, and it also reviews a number of challenges in this context, such as computational cost and lack of interoperability, which will require further research efforts. However, recently, research advances have made it possible to implement practical homomorphic cryptosystems, at least in Mobile environments (Ren et al., 2021; Shafagh et al., 2017). In particular, the encryption primitive used is the hash function and the operation we exploit is XOR. Homomorphic Hashing, first introduced by Bellare, Goldreich, and Goldwasser (Bellare et al., 1994) has been used for disparate application scenarios (Kim & Heo, 2012; Lewi et al., 2019; Yao et al., 2018). In particular, (Kim & Heo, 2012) proposes a device authentication protocol for smart grid systems based on the properties of this function to decrease the amount of computation on a smart meter. Whereas, the approach presented in (Yao et al., 2018) proposes a homomorphic hash and Blockchain-based authenticated key exchange in the context of social networks. Facebook researchers design a scheme based on Homomorphic Hashing to secure update propagation in the context database replication, ensuring consistency (Lewi et al., 2019).

In our approach, we leverage the properties of Homomorphic Hashing, in particular, related to the XOR operation, to allow the aggregator node, during the safe starting phase of our framework design, to identify groups of objects able to compute the device fingerprint of a target object, without revealing the identity of the target object itself. To the best

of our knowledge, the way we design this algorithm is novel and has never been used before.

A novel research direction to monitor the behavior of objects in IoT networks in a distributed way and provide some forms of trust or authentication is Blockchain (Ali et al., 2021; Chen et al., 2022; Dedeoglu et al., 2019; Hammi et al., 2018; Nofer et al., 2017; Pietro et al., 2018). In particular, the authors of (Pietro et al., 2018) present a framework based on the concept of *Islands of Trust*, that are portions of the IoT network where trust is managed by both a full local PKI and a Certification Authority. Service Consumers generate transactions forming an Obligation Chain first locally accepted by Service Providers and, then, shared with the rest of the network. Also the work presented in (Hammi et al., 2018) exploits a similar concept of secure virtual zones (called bubbles) obtained through Blockchain technology, where objects can identify and trust each other. Both the work presented in (Corradini et al., 2022; Dedeoglu et al., 2019) try to overcome Blockchain limitations proposing a light architecture for improving the end-to-end trust making this technology feasible to limited IoT devices. The proposal illustrated in (Dedeoglu et al., 2019) leverages some gateway nodes calculating the trust for sensor observations based on some parameters, such as: nodes reputation, data received from neighboring nodes, and the observation confidence. to compute the trustworthiness of a node, if the neighboring sensor nodes are associated with different gateway nodes, then, the gateway nodes are in charge of computing and sharing the evidence with their neighbors' gateway nodes. This architecture is not fully distributed and secure delegation is not performed; indeed, more powerful nodes are used as gateways. Whereas the work presented in (Corradini et al., 2022) describes a framework based on a two-tier Blockchain able to provide security and autonomy of smart objects in the IoT by implementing a trust-based protection strategy. This work leverages the idea of communities of objects and relies on a first-tier Blockchain to record transactions evaluating the trust of an object in another one of the same community or of a different community. After a certain time interval, these transactions are aggregated and stored in the second-tier Blockchain to be globally available. In our approach the use of Blockchain technology is limited to keeping trace of: *(i)* the identity of the device in charge to act as an aggregator for a target node; *(ii)* the evaluation of the behavior of aggregator after the aggregation task to enable the aforementioned FL approach; and *(iii)* the identity of objects for the anomaly detection task. Hence, differently from the above-cited approaches, the core of the strategy is not performed through Blockchain.

Another functionality provided by this paper is the possibility for the less capable devices to benefit and participate in our FL approach through secure delegation. This algorithm has been mentioned in the H2O framework (Ferretti et al., 2021), without developing a detailed implementation of it. Thanks to this paradigm, the training and inference phases of our model can be obtained through a privacy-preserving collaborative delegation approach in which power devices cooperate and provide support to less powerful ones to implement the solution without revealing the features of the model.

In the following, we summarize the comparison with the most important works introduced above based on the different functionalities provided by our approach, namely:

- Anomaly Detection: a capability to identify action sequences that deviate significantly from the expected behavior.
- Reputation Model: a functionality that allows a node in the network to compute a reliability score of another node based on trust values and according to its neighbors' opinion, even if they have not been in contact before.
- Privacy: the implementation of measures and strategies to protect the identity of the node during the computation of behavioral fingerprint models.
- Secure Delegation: a mechanism allowing devices to delegate tasks to more capable peers, by preserving the privacy of the involved nodes' identity.

With the letter 'x' we denote that the corresponding property is provided by the cited paper (Table 1).

3 Description of Our Approach

This section is devoted to the description of our proposal. In particular, in the next subsections, we provide a general overview of our approach along with its underlying model; we illustrate our Secure Multi-party computation strategy to form groups of co-workers for an FL task; after that, we detail our FL-based behavioral fingerprinting solution; finally, we sketch the adaption of an existing reputation model into our scenario.

3.1 General Overview

This section details the architectural design of our FL-based approach. In particular, we will describe the system actors and how they interact with each other during the model training and evaluation processes. Table 2 reports all the abbreviations and symbols used throughout this paper.

As typically done in the literature, our model for the considered IoT scenario is based on a directed graph $G = \langle N, E \rangle$, where N is the set of nodes and E is the set of edges representing relationships between pairs of nodes. In particular, a link is built if two nodes got in touch in the past exchanging one or more messages. Observe that the direc-

Table 1 Comparison of our approach with related ones

Approach	Approach Type	Anomaly	Reputation	Privacy	Secure
(Bezawada et al., 2018; Celdrán et al., 2022; Oser et al., 2018; Radhakrishnan et al., 2014)	Fingerprint	x	-	-	-
(Aramini et al., 2022)	Fingerprint	x	-	-	x
(Preuveneers et al., 2018; Rey et al., 2022)	FL, Blockchain	x	-	-	-
(Nguyen et al., 2019)	Fingerprint, FL, Blockchain	x	-	-	-
(Kim & Heo, 2012)	HE	x	-	x	-
(Yao et al., 2018)	HE, Blockchain	x	-	x	-
Dedeoglu et al. (2019); Hammi et al. (2018); Pietro et al. (2018)	Blockchain	x	x	-	-
Ferretti et al. (2021)	Fingerprint, Consensus	x	x	-	-
Our approach	FL, Fingerprinting				
	Consensus, Delegation, HE	x	x	x	x

Table 2 Summary of the main symbols and abbreviations used in our paper

Symbol	Description
FL	Federated Learning
SMC	Secure Multi-party Computation
HE	Homomorphic Encryption
N	The set of IoT nodes of the network
N_l	The set of basic devices, a subset of N
N_m	The set of devices with medium computation power, a subset of N
N_p	The set of powerful devices, a subset of N
$\|N\|$	Cardinality of the set of IoT nodes
n_i	An IoT device of N
c_i	A worker device of N
id_{c_i}	The id of a worker device c_i
b	A target node
a_b	An aggregator node for b
id_{a_b}	The id of an aggregator a_b
Λ_b	List of workers training a model on b
Γ_n	the set of neighbor nodes of n
$\mathcal{H}$	Homomorphic Hash Function
η, ξ	Nonces
t	The size of a sequence of input symbols of the deep learning model
d_i	Delegate node of c_i for a task
th_w	Threshold for mispredicted symbols
$T_{c_i,b}$	Trust score assigned by a node c_i towards a target node b
$\mathcal{FP}_{w_b}$	Behavioral fingerprinting function of b during an observation window w_b
R_b^{ω}	Reputation of b after each time period ω
τ	Tolerance value
m	A generic Machine Learning evaluation metric
ϕ_{ban}	Ban interval

tion of the link identifies the node that starts to communicate during the message exchange. The group of peers a node n_i has been interacting with is the set of neighbors of n_i and can be defined as $\Gamma_{n_i} = \{n_j \in N : (n_i, n_j) \in E\}$.

Moreover, in our model, N is partitioned into three subsets according to the different object capabilities, thus resulting in $N = N_p \cup N_m \cup N_l$. The subset of powerful devices N_p includes all the devices with sufficient capabilities in terms of memory and computational strength to perform the more demanding tasks of our approach (e.g., the training ML/DL models). The second set N_m is composed of devices with medium computational and memory capabilities, due to their battery constraints or power stability. The last set N_l is composed of less capable nodes with basic functionalities. Since they have limited computational power, they can rely on delegation to more powerful nodes to participate in our framework.

As stated in the Introduction, the proposal described in this paper focuses on the computation of behavioral fingerprinting models via FL. To do so, our strategy assumes the existence of an initial phase, called the *safe starting phase*, in which several actors can train ML/DL models to learn the behavior of target nodes in an environment free from possible attacks to these targets (i.e., no attacks are performed on any involved target node capable of altering its behavior). During this phase, IoT nodes can play one of the following roles:

- *Worker*. It is in charge of training a local behavioral fingerprinting model of a target node. Since training such a model is the more demanding task in our solution in terms of computational and memory capability, these nodes belong to the N_p set.
- *Aggregator*. They are in charge of aggregating the local contributions of the different workers of an FL task to compute a global model for a target. This task is less computationally demanding than the previous one, hence it can be taken over by nodes belonging to $N_p \cup N_m$ (See Section 5.2 for details on the performance).
- *Target*. They are the monitored nodes for which the behavioral fingerprinting has to be computed. There are no requirements in terms of computational power for them, hence they can belong to any subset of nodes defined above ($N_p \cup N_m \cup N_l$).

During this phase, less and medium-capable nodes belonging to $N_m \cup N_l$ can participate in the scheme leveraging a secure delegation approach. In particular, they can entrust nodes in N_p to carry out actions on their behalf exchanging data in a privacy-preserving way. The details of this task are described in Section 3.3.

Subsequently, in the *fully operational phase*, also referred to as *inference* phase, learned models are used by all the actors to infer possible anomalies on the monitored targets. This phase is less impacting than the training one in terms of computational requirements, hence all the objects belonging to $N_p \cup N_m$ can actively participate in this phase. It is worth noting that, also during this phase, less capable nodes belonging to N_l can entrust nodes in $N_p \cup N_m$ for the inference of behavioral fingerprinting models, through the aforementioned secure delegation strategy.

The last actor of our approach is the Blockchain. This technology provides a shared ledger to record trusted information accessible to all the nodes over the network. In particular, we leverage smart contracts running on the Blockchain to automatically execute predefined actions when certain conditions are met. Since smart contracts are stored on the Blockchain, their code and execution history are visible to all participants in the network enhancing transparency in transactions. In particular, we leverage this paradigm to keep track of several aspects, namely:

- The information necessary to discover the identity of aggregators for target nodes. In our approach, neither the workers know each other nor the aggregator knows the identity of the target. For these reasons, we design our framework to include Blockchain technology, thus removing the need of a trusted central authority or counterpart to keep information private.
- The trust scores assigned by workers to estimate the reliability of an aggregator. As a matter of fact, the use of Blockchain for this task enhances trust and prevents manipulation of scores. Through smart contracts', code is executed automatically to compute these complex measures starting from trust scores.
- The identity of corrupted objects resulting from the monitoring activity of nodes owing behavioral fingerprint model towards target peers. Once our anomaly detection framework has detected a change in the behavior of a node, it is important to publish this information in an immutable and trusted ledger accessible by every node of the network.

Figure 1 shows the general architecture of our solution illustrating the different actors of the model. In particular, c_1, c_2, c_3 are three worker nodes, b is the target node, and a_b is the aggregator for b. The right part of this figure shows the Blockchain exploited during a number of steps of our approach. It is worth noting that, the interactions between the aggregator and the workers take place only during the *safe starting phase* to train the behavioral fingerprinting model of the target. In the subsequent phase, nodes communicate with each other and can leverage both trained models and the information stored in the Blockchain to evaluate the behavior of a contact. It is worth observing that, in our scenario, an anomaly in the behavior of a node can be caused

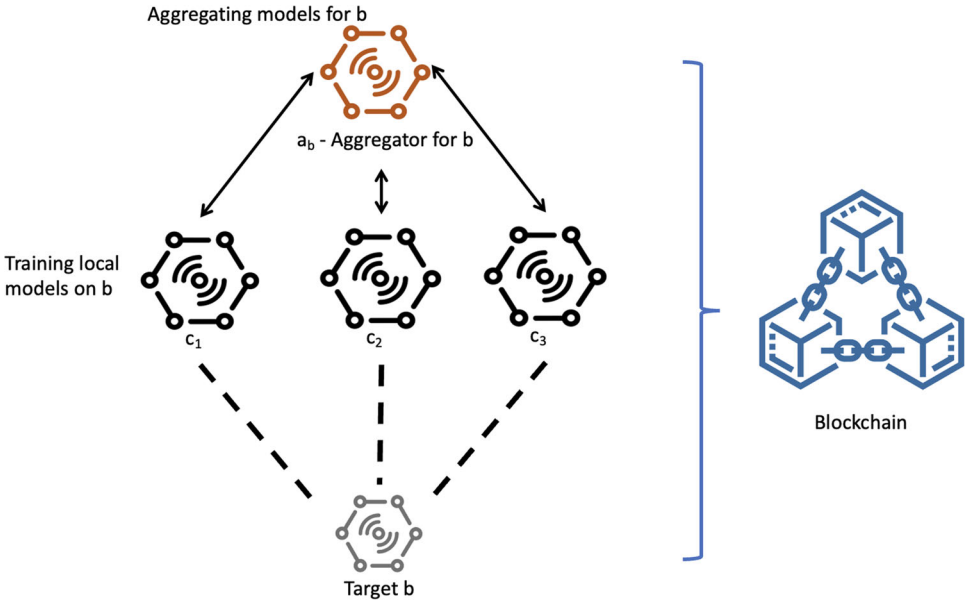

Fig. 1 The general architecture of our solution

by either a hardware malfunction, an environmental change, or an ongoing cyber attack. For the estimation of a change in the observed node behavior, a true positive will be signaled if the number of unexpected actions as predicted by our models exceeds a certain threshold. This happens also in the case of some external causes (like environmental changes). Moreover, our strategy leverages a mechanism to estimate trust scores on the basis of the detected behavioral anomalies and compute nodes' reputations. If the reputation of a node, computed by aggregating all the trust contributions towards it, goes under a reference threshold, it will be isolated by the other peers and, therefore, it is technically banned from the system (for, at least, a time ϕ_{ban}). At this point, system administrators can decide to restore the node or retrain its behavioral fingerprinting models, especially if the external cause is known and under control.

In the next sections, we will describe our approach in detail.

3.2 A Secure Multi-Party Computation Strategy to Identify Federated Learning Co-Workers

This section is devoted to the definition of a privacy-preserving strategy to identify the correct aggregator for a specific target and, hence, define groups of workers that can collaborate on an FL task. As said above, in our approach, each FL task is focused on the construction of a behavioral fingerprinting model for a target node of the network.

In practice, given a target node b, the above reasoning involves two actions that must be carried out to configure the FL task: *(i)* the identification of the aggregator for a target node, and *(ii)* the creation of the group of workers for the subsequent training task. It is worth noting that these tasks are performed by keeping the identities of the involved actors private. To do so, we develop a privacy-preserving strategy for group formation and identity exchange based on a Secure Multi-party Computation (SMC) strategy.

It is important to underlying that, as stated above, the actions above are performed during a *safe starting phase*, in which no attacks occur against the target b. We assume that such a phase is admissible and, typically, it can coincide with the system setup period or any subsequent maintenance action involving b.

Given a node $c_i \in N_p$ aiming at learning the behavioral fingerprints of b. Let id_{c_i} be the identifier of c_i, and let η be a private nonce generated by b. Finally, let $\mathcal{H}()$ be a homomorphic hash function preserving the XOR operation (Lewi et al., 2019). Our solution would enforce the following steps.

First, c_i contacts b to exchange a message containing information about id_{c_i} and a nonce generated by b, say η. A suitable payload is generated by b crafting the identifier of c_i and η, through a bitwise XOR operation. The result of the XOR operation is transformed by b using the homomorphic hash function, thus obtaining the final payload $\mathcal{H}(id_{c_i} \oplus \eta)$.

After receiving the first contact from c_i, b proceeds by identifying its referring aggregator. In our scenario, any node of $N_p \cup N_m$ can play the role of the aggregator, provided that it is associated with a sufficient trust score. The details concerning the trust mechanism are reported in Section 3.4. In any case, the eligible aggregators along with their trust scores are stored in the underlying Blockchain. Once b has identified its aggregator a_b, it will create a new transaction in the Blockchain to publish this information. However, our solution requires that the association between b and a_b can only be disclosed by b to the nodes it wishes to involve in

the subsequent FL task. This would confer to b the capability of filtering out unwanted workers from the learning task of its behavioral fingerprinting model. To do so, b computes a secret by applying again the homomorphic hash function to a payload composed of the bitwise XOR between the public identifier of the selected aggregator id_{a_b} and its private nonce η. Consequently, the public transaction on the Blockchain generated by b does not save the plain identifier of its aggregator, but the secret $\mathcal{H}(id_{a_b} \oplus \eta)$.

At this point, when c_i wants to gather the identity of the aggregator selected by b, it will retrieve the transaction generated by b from the Blockchain, containing $\mathcal{H}(id_{a_b} \oplus \eta)$, and it will carry out the following computation. First, it performs a bitwise XOR operation with: *(i)* the hash received by the target, namely $\mathcal{H}(id_{c_i} \oplus \eta)$; *(ii)* $\mathcal{H}(id_{a_b} \oplus \eta)$; and *(iii)* the hash of its own identity $\mathcal{H}(id_{c_i})$. For the properties of homomorphic hashing concerning the XOR operation, we have the following equation:

$$\begin{aligned} &\mathcal{H}(id_{c_i} \oplus \eta) \oplus \mathcal{H}(id_{a_b} \oplus \eta) \oplus \mathcal{H}(id_{c_i}) \\ &= \mathcal{H}(id_{a_b} \oplus \eta) \oplus \mathcal{H}(\eta \oplus id_{c_i} \oplus id_{c_i}) \\ &= \mathcal{H}(id_{a_b} \oplus \eta) \oplus \mathcal{H}(\eta) \\ &= \mathcal{H}(id_{a_b} \oplus \eta \oplus \eta) \\ &= \mathcal{H}(id_{a_b}) \end{aligned} \tag{1}$$

Now, c_i can retrieve from the Blockchain the list of available aggregators. For each identifier in such a list, c_1 can apply $\mathcal{H}()$ to it and compare the result with the value from the previous computation. The search for the correct aggregator will be completed when a match is found. Algorithm 1 summarizes the steps above for the privacy-preserving discovery of id_{a_b}. Observe that, the computational complexity of such an algorithm is $\mathcal{O}(|L|)$, where $|L|$ is the number of possible aggregators in the system.

After this step, c_i is now equipped with the identity of the aggregator a_b for the target b, hence c_i is ready to contact a_b to notify its intention to train a model for b.

The steps carried out by c_i are repeated by any other node c_j of N_p interested in a model for b. Our solution does not enforce any restriction on the number of FL tasks an aggregator could be involved in. Indeed, as will be shown in Section 5, the computational complexity required for the aggregation is not very high and, therefore, can be easily executed by any node of $N_p \cup N_m$. However, a_b must identify and synchronize all the nodes related to a specific FL task (i.e., a task dedicated to a given target b). Again, our solution enforces that a_b must not know the identity of b and, therefore, the identification of the groups of workers can be performed as shown in Algorithm 2. In particular, given a list of nodes $\Gamma_{a_b} = \langle c_1, c_2, ..., c_n \rangle$ that contacted the aggregator a_b, the identification of the groups of workers is done through an iterative algorithm. For each worker $c_i \in \Gamma_{a_b}$ the aggregator

Algorithm 1 Discovering Aggregator identity

Data: $c_i \in N_p, b \in N, a_b \in N_p \cup N_m$; /* node, target node, aggregator node for b */
$\eta, \mathcal{H}$; /* nonce of b, homomorphic hash function for XOR */
$L = \{id_x, x \in N\}$; /* list of the aggregator identifiers in the Blockchain */
$\mathcal{H}(id_{a_b} \oplus \eta)$; /* secret for the aggregator of target node in the Blockchain */
Result: id_{a_b}
c_i contacts b;
$c_i \leftarrow \mathcal{H}(id_{c_i} \oplus \eta)$ from b;
c_i computes $\mathcal{H}(id_{c_i})$;
c_i computes $\widetilde{\mathcal{H}}(id_{a_b}) = \mathcal{H}(id_{c_i} \oplus \eta) \oplus \mathcal{H}(id_{a_b} \oplus \eta) \oplus \mathcal{H}(id_{c_i})$;
foreach $id_x \in L$ **do**
  c_i computes $\mathcal{H}(id_x)$;
  if $\mathcal{H}(id_x) == \widetilde{\mathcal{H}}(id_{a_b})$ **then**
    $id_{ab} = id_x$
  end
end

computes the hash of its identity $\mathcal{H}(id_{c_i})$ and performs a bitwise XOR operation with the secret previously received from c_i (i.e., $\mathcal{H}(id_{c_i} \oplus \eta)$). Due, once again, to the homomorphic property of the hash function for the bitwise XOR, this will result in the following.

$$\Xi_{c_i} = \mathcal{H}(id_{c_i}) \oplus \mathcal{H}(id_{c_i} \oplus \eta) = \mathcal{H}(id_{c_i} \oplus id_{c_i} \oplus \eta) = \mathcal{H}(\eta)$$

Now, for each other node $c_j \in \Gamma_{a_b} \setminus c_i$, the aggregator performs a XOR operation between Ξ_{c_i} and the secret previously received by c_j, say $\mathcal{H}(id_{c_j} \oplus \eta')$. Thus obtaining:

$$\Xi_{c_i} \oplus \mathcal{H}(id_{c_j} \oplus \eta') = \mathcal{H}(\eta) \oplus \mathcal{H}(id_{c_j} \oplus \eta') = \mathcal{H}(\eta \oplus id_{c_j} \oplus \eta')$$

Now, if $\eta = \eta'$ holds, then the previous computation will be equal to $\mathcal{H}(id_{c_j})$. Since we assumed that different targets will always have different nonces (no collision between generated nonces), this result would mean that c_i and c_j share the same target and, hence, they belong to the same working group Λ_b. Observe that, a_b can directly compute $\mathcal{H}(id_{c_j})$ for c_j to verify the equality between the results of the computation above and the identifier of c_j. The computational complexity for the group identification algorithm is $\mathcal{O}(\Gamma_{a_b})$, where Γ_{a_b} is the number of nodes that contacted a_b for an aggregation task.

The sequence diagram in Fig. 2 summarizes all the steps performed during the *safe starting phase* of our approach.

3.3 Distributed Behavioral Fingerprinting via Federated Learning

This section is devoted to the description of the Federated Learning strategy for the computation of behavioral

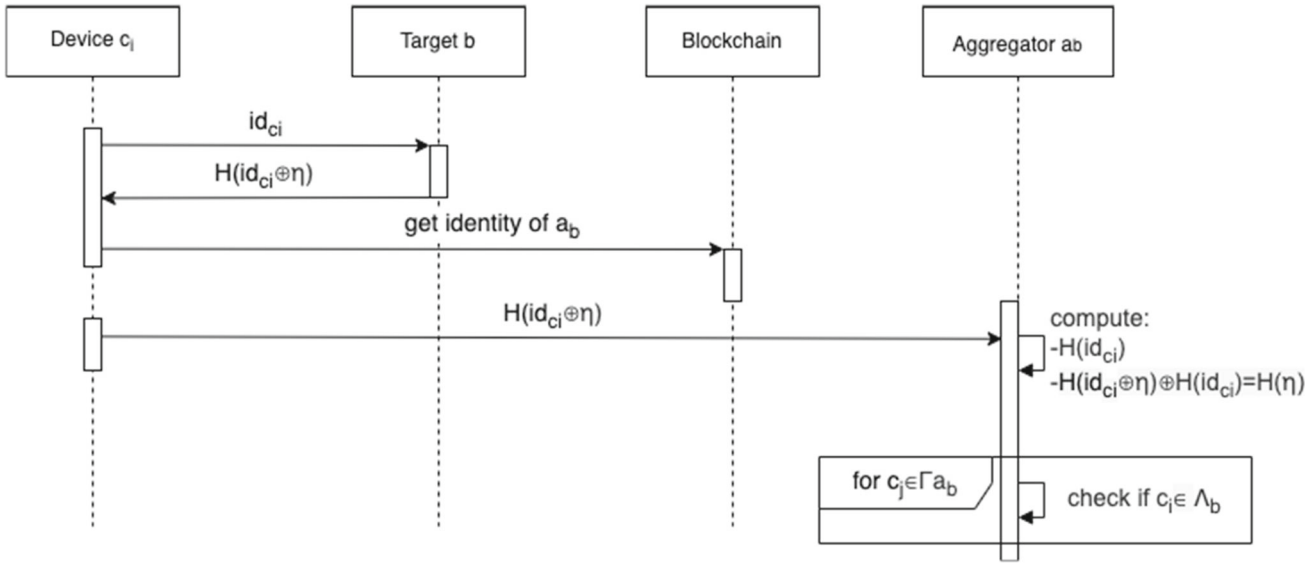

Fig. 2 The sequence diagram of all the FL setup steps performed during the *safe starting phase* of our solution

fingerprinting models. Practically speaking, FL is a distributed collaborative machine learning approach that allows algorithm training across multiple decentralized devices holding local data samples without sharing the actual datasets (Konečnỳ et al., 2015). Recently, this paradigm has been investigated for building intelligent and privacy-enhanced IoT applications (Nguyen et al., 2021; Sánchez et al., 2021).

Although few works leverage this strategy for anomaly detection in IoT, they are focused on building classical device fingerprints based on basic parameters, like usage of CPU, memory, and so on (Sánchez et al., 2022, 2021). The novelty of our contribution concerns the fact that we aim to construct a global device behavioral profile taking into account all the interactions over the network, even across different services, a node may provide.

Consider, for instance, the example shown in Fig. 3 about a smart thermostat. This device can detect multiple metrics, such as the temperature and humidity of the room in which it is located; it can connect to other smart devices via Bluetooth or directly to the Internet allowing the owner to monitor the home situation, remotely. Moreover, it can control the home heating system according to the detected temperature. Finally, it could also communicate with a central home alarm system in the case in which a fire or anomaly temperatures have been detected. Hence, this device holds interfaces with the actors it interacts with, providing different services to each of them. This means that the communications and the messages it exchanges can be very different according to the service it is providing.

Classical decentralized behavioral fingerprinting solutions (Aramini et al., 2022; Bezawada et al., 2018; Ferretti et al., 2021) consider only a single interaction sequence to build a profile of a target node and they neglect a comprehensive point of view coming from the messages exchanged between the target and its other neighbors. Hence in the example shown above, the home heating system will build an ML model of the thermostat, which will differ from the one trained by the home alarm system or any other smart device.

Our strategy leverages FL to build behavioral fingerprinting models combining the perspectives of different workers (neighbors of a target node) in a global profile. Ultimately,

Algorithm 2 Training groups identification

Data: $c_i \in N_p, b \in N, a_b \in N_p \cup N_m$; /* node, target node, aggregator node for b */
$\eta, \mathcal{H}, \Gamma_{a_b}$; /* nonce of b, homomorphic hash function, set of nodes that contacted a_b */
$\Phi_{\Gamma_{a_b}} = \{\mathcal{H}(id_{c_j} \oplus \eta') : c_j \in \Gamma_{a_b}\}$; /* The set of secrets sent by the nodes of Γ_{a_b} to a_b */
Result: $a_b \leftarrow \Lambda_b$; /* List of nodes that will train a model on b */
$c_i \in \Lambda_b$;
$c_i \rightarrow \mathcal{H}(id_{c_i} \oplus \eta)$ to a_b;
a_b computes $\mathcal{H}(id_{c_i})$;
a_b computes
$\mathcal{H}(id_{c_i}) \oplus \mathcal{H}(id_{c_i} \oplus \eta) = \mathcal{H}(id_{c_i} \oplus id_{c_i} \oplus \eta) = \mathcal{H}(\eta)$;
foreach $c_j \in \Gamma_{a_b}$ **do**
  a_b computes $\mathcal{H}(\eta) \oplus \mathcal{H}(id_{c_j} \oplus \eta') = \mathcal{H}(\eta \oplus id_{c_j} \oplus \eta')$;
  a_b computes $\mathcal{H}(id_{c_j})$;
  if $\mathcal{H}(\eta \oplus id_{c_j} \oplus \eta') = \mathcal{H}(id_{c_j})$ **then**
    $c_j \in \Lambda_b$
  end
end

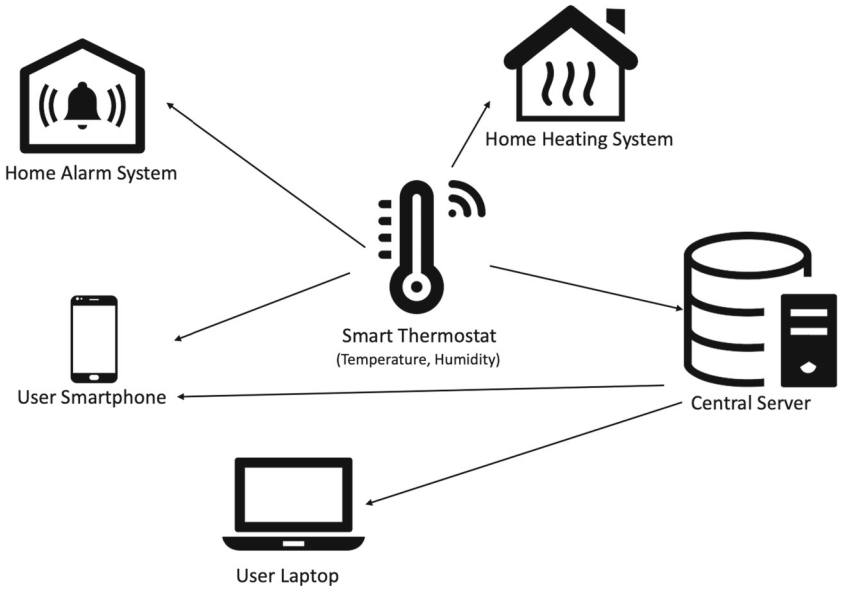

Fig. 3 Smart Thermostat interactions in a domotic environment

this would depict the behavior of the target device in a more general way.

Nevertheless, the global model is fed with the single interaction sequences, for which we leverage an adaptation of the behavioral fingerprinting solution described in (Aramini et al., 2022). Observe that, according to our fully distributed architecture, a worker has always access to payload data as it is the intended recipient of the communication with the target. Therefore, we can follow the solution described in (Aramini et al., 2022), thus including payload-based features in our strategy. These additional features allow also for the protection against cyber-physical attacks, in which an attacker tries to jeopardize sensing data to alter the behavior of the cyber-physical environment. In addition to payload-based features, to characterize the behavior of an object this approach considers also classical network parameters (i.e., source port type, TCP flag, encapsulated protocol types, the interval between arrival times of consecutive packets, and packet length) altogether with features derived from the payload. Then it proceeds by mapping the sequence of exchanged packets in a sequence of symbols and leverages a Gated Recurrent Unit (GRU) neural network composed of 2 layers of 512 and 256 neurons, respectively, a fully connected layer with size 128, and an output classification layer. The choice of a GRU as the reference model, instead of more complex architectures (such as LSTM), is due to the need of solving the trade-off between the solution accuracy and the computational complexity of training behavioral fingerprinting models for IoT nodes. The objective of the deep learning model is to classify the next symbol given a sequence of input symbols of size t[1].

In the remainder of this section, we illustrate how we apply FL in our approach. In the previous sections, we focused on the description of the setup tasks crucial for the privacy-preserving execution of our scheme, namely: *(i)* the identification of the aggregator device for a target node, and *(ii)* the creation of groups of workers for FL training task. At this point, since all the roles have been assigned, the aggregator first initializes a global model with random learning parameters. Secondly, each worker gets in contact with the aggregator to receive the current model and, after this step, it computes its local model update. To do so, each node leverages its own dataset gathered from the direct interaction sequence with the target node. At each training epoch, once the local contribution is computed, the worker can forward it to the aggregator that is in charge of combining all the local model updates and, hence, it constructs an enhanced global model with better performance, still ensuring protection against privacy leakages. The last two steps are performed iteratively until the global training is complete.

Figure 4 sketches the steps described above focusing on the communication between one of the involved workers and the aggregator.

3.3.1 Leveraging Secure Delegation

It is worth observing that, because of the high heterogeneity of devices in an IoT network, not all the nodes are equipped with sufficient computational and memory capability to execute the training phase of our approach. Hence, we resort to a secure delegation mechanism according to which less powerful devices in $N_l \cup N_m$ can delegate such tasks to powerful

[1] Observe that, the value of t can be fixed based on the dynamicity of the object-to-object interactions. In our experiment (see Section 5) following the results described in (Aramini et al., 2022), we set this value to 10.

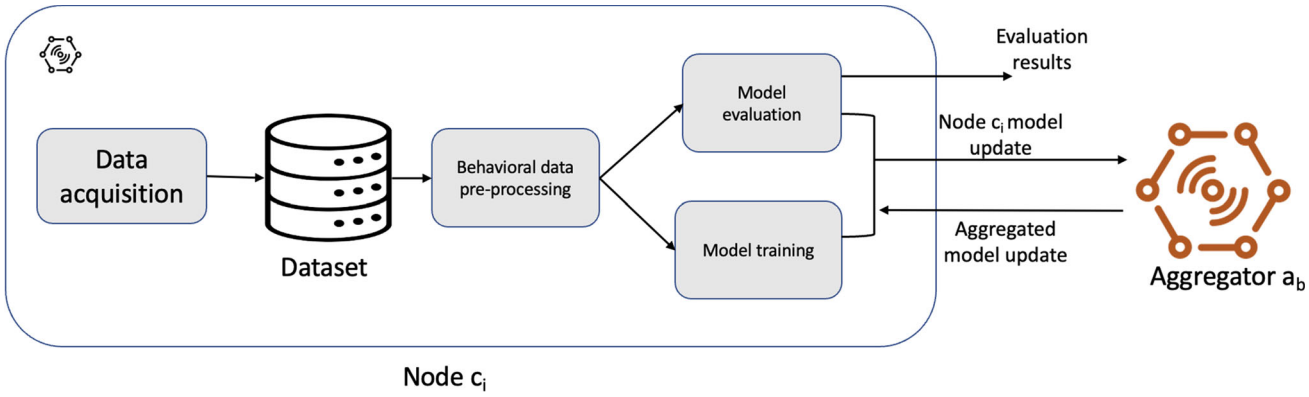

Fig. 4 Detailed view of the interaction between a worker and the aggregator during the training of a FL model

devices in N_p. In the recent literature, some theoretical models and ontologies have been designed for the identification of reliable IoT devices for secure delegation, tackling the issue of incomplete task requests owned by resource-constrained IoT devices (Khalil et al., 2021). Of course, any existing secure delegation strategy could be adopted in our approach. However, for the sake of completeness, we describe a naive approach in which both the training and the subsequent inference phases can benefit from delegation.

In particular, in the following, we describe the two scenarios above, separately. We start with the training phase and we consider the situation in which a less capable device, say c_i, is involved as a worker in the construction of a behavioral fingerprinting model for a target b. We assume that, due to the lightweight nature of the operations described in Section 3.2, any node can perform the setup steps for the configuration of the FL task (see the experiments on the performance of IoT nodes on these tasks in Section 5.2). In practice, c_i can execute both Algorithms 1 and 2 to identify the aggregator for b and become a member of the working group to build its behavioral fingerprinting model. Secure delegating is, hence, needed in the subsequent steps involving the training of the local ML model.

According to our strategy, given a cryptographic salted hash function $\mathcal{H}_f(v, s)$ (Rana et al., 2022), in which v is the value to be hashed and s is the salt, the secure delegation of the training phase requires the following steps:

- collection of interaction packets with the target b;
- feature extraction and mapping with the corresponding symbols (as described before);
- pre-processing of the symbol sequence to guarantee privacy;
- upload of the training set in a shared data bucket linked in the Blockchain;
- identification of a trusted delegated node in the network;
- interaction with the delegated node to start the training.

First, c_i collects a sequence of interaction packets during its communication with b. Adopting the approach described in (Aramini et al., 2022), it, then, extracts both payload-based and network-based features from such a sequence. It, then, maps each unique combination of these features to a corresponding symbol. At this point, a sequence of interaction packets is replaced by a sequence of symbols.

Now, without losing information, to protect the privacy of the communications between the worker c_i and b, our approach imposes that each symbol of such a sequence can be converted into its hash representation using the salted secure hash function mentioned above. In this way, only the source node c_i can know the mapping between the original symbol sequence and the hashed one. This facility is enabled at the FL task level, i.e. once a node c_i expresses its need for a secure delegation, the whole FL task will be adjusted to work with a converted set of symbols. To do so, c_i communicates its need to use secure delegation to the aggregator a_b. The latter will, then, generate a salt s that will be sent to all the workers involved in the FL task having b as a target. At this point any packet sequence $\langle pkt_1, pkt_2, \cdots, pkt_m \rangle$ will be converted, first into a sequence of symbols according to the values of the considered features of each packet, namely $\langle sy_1, sy_2, \cdots, sy_m \rangle$. Then, each node will apply the secure salted hash function $\mathcal{H}_f$ to obtain the hashed symbol sequence $\langle \mathcal{H}_f(sy_1, s), \mathcal{H}_f(sy_2, s), \cdots, \mathcal{H}_f(sy_m, s) \rangle$. Observe that, while the first transformation can be done by any node in the network and, hence, knowing a sequence of symbols it is possible to derive information about the original packet sequence, due to the property of the adopted cryptographic salted hash function, it is not possible to invert the hashed symbol sequence into its original packet sequence. As a consequence, only the nodes involved in the FL task, which know the salt s, can obtain the hashed symbols from a sequence of packets and, hence, exploit the trained model.

As for the identification of a trusted delegated node, our approach can leverage any existing state-of-the-art trust

model for IoT. In Section 3.4, we provide an overview of a possible trust scheme and extend it to include support for the identification of aggregators. The only requirement is that c_i can estimate the reliability of its peers so as to identify the correct delegate d_i for its task.

At this point, c_i can share its privacy-preserving training set with d_i to start the training phase. To do so, we leverage IPFS as a global file system in which nodes can upload their data. Moreover, the links to IPFS folders are shared through transactions on the Blockchain. Of course, our privacy-preserving strategy does not require additional security mechanisms on IPFS to protect the training set. Indeed, as stated above, any node in the network could use these data to train a model, however, only the node involved in the specific FL task will know the salt s and, hence, can perform the mapping between the hashed symbol sequence and the real packet one. With that said, d_i can carry out the training task for c_i by receiving the initialized global model of the FL task from it. At each epoch, d_i will return the local model updates to c_i and it will receive the updated global model for the following training epoch.

After the training phase, c_i will receive the final version of the trained model from a_b. However, if the delegation embraces also the model inference, the delegated node retains the trained model to support c_i also for model inference.

In particular, the secure delegation for the inference phase works as follows. First, c_i collects the packet sequence from its direct interaction with b. Then, it converts this sequence into the corresponding symbol sequence and, hence, applies $\mathcal{H}_f$, using the same salt s obtained by a_b during the training phase, to build the hashed symbol sequence. This last can, then, be used by d_i as input to the trained behavioral fingerprinting model.

3.3.2 Exploiting behavioral fingerprints for Anomaly Detection

The steps described above focus on the creation of deep learning models that, given an input symbol sequence, are capable of classifying its next symbol. The advantage brought about by our solution is that to estimate the behavior of a node, it considers not only a single point-to-point interaction between two peers, but a community-oriented general perspective of the target node. However, although the performance of such a classifier is extremely high as will be shown in Section 5, using a single prediction to identify a change in the behavior of a node is not adequate and could lead to false predictions. To avoid this issue, as suggested by the related literature (Aramini et al., 2022; Nguyen et al., 2019), we adopt a window-based strategy. Specifically, given an observation window, say w_b, our approach exploits the aforementioned classifier to identify mispredicted symbols. As for the estimation of a change in the observed node behavior, a true positive will be signaled if the number of mispredicted symbols exceeds a threshold th_w. Such a threshold should be suitably tuned to dampen the, even low, false prediction rate of the underlying classifier. Practically speaking, if the overall confidence of the classifier is 0.80, to dampen the prediction errors, th_w should be fixed to a value greater than 20% of the window size. Of course, the choice of the correct value for th_w, although its lower bound can be established by the reasoning above, strongly depends on the dynamics of the IoT scenario under analysis. Indeed, a greater th_w implies a slower detection of behavior changes for the target nodes (Aramini et al., 2022).

3.4 The Underlying Trust Model

In this section, we sketch the underlying trust model exploited by our solution. Indeed, in the previous sections, we stated that an IoT node can select suitable aggregators and/or delegated nodes by leveraging the information stored in the Blockchain about node reliability. Behavioral fingerprinting can be a key factor in the construction of enhanced reputation models. Indeed, it can be used to estimate anomalous actions that can be grounded on security attacks or device malfunctions. The definition of a model to estimate trust scores and compute nodes' reputations is an orthogonal study concerning our approach; therefore, to build our solution, we can leverage existing proposals to provide forms of trust in an IoT network (Corradini et al., 2022; Dedeoglu et al., 2019; Pietro et al., 2018).

In particular, in our proposal, we adopt the approach of (Corradini et al., 2022) to estimate trust and reputation scores. In the following, we briefly sketch the adaptation of such an approach into our application scenario. Specifically, in our context, a trust score can be assigned by a node c_i towards a target node b, for which it holds a behavioral fingerprinting model, as follows:

$$T_{c_i,b} = 1 - \mathcal{FP}_{w_b}(c_i, b)$$

Here, $\mathcal{FP}_{w_b}$ is a function that exploits the behavioral fingerprinting model of b to estimate changes in its behavior during an observation window w_b. This function can naively record the number of mispredictions registered during w_b and compute the ratio between such a number and the total length of the packet sequence exchanged by c_i and b during w_b. As done in (Corradini et al., 2022), such trust scores can be published by the monitoring node c_i in the Blockchain. Therefore, given a fixed time period $\omega > w_b$, let TS_b^{ω} be the set of trust transactions published by any node holding

a fingerprinting model towards b. Moreover, let $\overline{T_b^\omega}$ be the average trust score in TS_b^ω. The reputation after each time period ω can be computed as follows.

$$R_b^\omega = \begin{cases} \alpha \cdot R_b^{\omega-1} + (1-\alpha) \cdot \overline{T_b^\omega} & \text{if} |TS_b^\omega| \neq 0 \\ R_b^{\omega-1} & \text{otherwise} \end{cases}$$

In this equation, again as stated in (Corradini et al., 2022), α is a parameter introduced to tune the importance of past behavior observations concerning new ones.

As an additional trust contribution, we design a specific trust score for aggregators. An aggregator can be also evaluated based on its honesty in constructing global models during FL tasks. To do so, we introduce an additional check that the involved workers can perform during the training epochs. Given a normalized performance metric m, at each epoch e, a worker c_i can compare the value of m for the local model, say m_l, and for the global one returned by the aggregator a_b for this epoch, namely m_g. In practice, such an additional trust score can be formulated as follows.

$$T_{c_i,a_b} = |m_l - m_g| \cdot (1-\tau)$$

Here, τ is a tolerance value introduced to absorb the expected variations in the values of the chosen metric between the global and local models. Finally, as for the metric m, it can be any evaluation metric typically adopted for machine learning models, such as the *accuracy*, the *prevalence*, the *f-measure*, and so forth.

4 Security Model

This section is devoted to the security model underlying our solution. In particular, we introduce both the attack model and the security analysis proving that our approach is robust to possible attacks.

4.1 Attack Model

We start this section with a preliminary assumption according to which our approach is applied to a scenario already in a stationary situation, or *fully operational phase*, with enough nodes available to carry out all the steps required by our scheme. For this reason, we do not consider the initial start-up stage, which can be characterized by an IoT network not yet active or complete. Moreover, as stated in Section 3, we assume the existence of a *safe starting phase* in which the nodes are configured and the behavioral fingerprinting models can be trained.

In the following, we list the assumptions useful for analyzing the security properties of our model.

A.1 There exists an initial safe phase in which behavioral fingerprinting models are built in the absence of attacks on target nodes.
A.2 An attacker cannot control the majority of the workers by training a behavioral fingerprinting model associated with a target.
A.3 An attacker has no additional knowledge derived from any direct physical access to IoT objects.
A.4 The exploited Blockchain technology is compliant with the standard security requirements commonly adopted for Blockchain applications.
A.5 The nonces and identifiers of nodes are generated starting from different key spaces. Moreover, no pair of identifiers or nonces can collide.

As stated above, our model ensures a list of security properties (SP, in the following), as follows:

SP.1 Resistance to attacks on Federated Learning.
SP.2 Resistance to attacks on the SMC strategy to identify FL co-workers.
SP.3 Resistance to attacks on the Blockchain and the Smart Contract technology.
SP.4 Resistance to attacks on the Reputation Model.
SP.5 Resistance to attacks on the IoT network.

4.2 Security Analysis

This section presents the analysis of the security properties listed above to prove that our approach can ensure them. In the following, we provide a detailed description of such analysis for each of the properties listed above.

4.2.1 SP.1 - Resistance to attacks on Federated Learning

Our approach leverages Federated Learning during the *safe starting phase* in which the behavioral fingerprinting models have to be trained for target nodes. For Assumption A.1, during this stage models computation is performed in the absence of attacks against target nodes. However, both the workers and the aggregator nodes can be forged or attacked. As for the first case, the large threat surface of the Federated Learning scenario makes this new type of distributed learning system vulnerable to many known attacks targeting worker nodes (Jere et al., 2020). In general, these security attacks focus on poisoning the model or preventing its convergence. In our approach, we can consider the protection against these attacks as an orthogonal task. Indeed, in the current scientific literature, there exist several countermeasures that FL aggregators can adopt to identify misbehaving workers and, hence, discard their contributions. Examples of such strategies are, for instance, the robust aggregation functions *AGRs*, such as *Krum*, *Trimmed Mean*, and so forth

(Blanco-Justicia et al., 2021). These represent lightweight heuristics that can be easily adopted in our scenario to provide robustness against common attacks.

Considering the second case in which the aggregator nodes are corrupted, our approach natively supports a countermeasure to possible attacks targeting them. Indeed, in Section 3.4, we include a facility in the underlying trust model to evaluate their honesty. The trust score, used to assess the quality of its aggregation behavior, is computed by analyzing the performance of partial local models and the global one generated by the aggregator during each epoch. If this value goes under a reference reliability threshold, the aggregator cannot be contacted by other nodes in the future. To avoid the permanent removal from the system of a node, we could hypothesize a ban interval, say ϕ_{ban}, after which the default reputation value will be restored. Of course, for critical scenarios, ϕ_{ban} can also be infinite. Therefore, no advantage is obtained by the attacker if, after a malicious behavior, the node is forbidden to interact with the network for a possibly long period.

4.2.2 SP.2 - Resistance to attacks on the SMC strategy to identify FL co-workers

In our scenario, during the phase related to the formation of the groups of workers for FL tasks (see Section 3.2), a malicious node can try to contact a victim node, say b, to discover its secret nonce η. Holding this value the attacker can infer the identities of the workers for the victim b. To do this, it performs a cryptographic attack exploiting the properties of HE. Indeed, it queries multiple times b trying to guess η and analyzing the result. In particular, it sends to b a value that is not its identifier but a guessing value for η, say η'. If it succeeds in the guessing of η (i.e. $\eta' = \eta$) b will return $\mathcal{H}(\eta' \oplus \eta) = 0$. At this point, the attacker can violate the SMC scheme and break our privacy-preserving algorithm. This attack can then be used to implement active eavesdropping, as a malicious node can sense the messages exchanged between two nodes and try to oust the intended target node to take some advantage.

This attack cannot happen thanks to the Assumption A.5, indeed the nonce and the identifier of the nodes have to be chosen in different key spaces. Therefore, an attacker cannot guess the nonce of the victim by forging a suitable identifier as shown above.

4.2.3 SP.3 - Resistance to attacks on the Blockchain and the Smart Contract technology

This category of attacks tries to exploit known vulnerabilities of the Blockchain and the Smart Contract technology. This new paradigm has been widely used in a variety of applications in recent years, but it still presents open issues in terms of security (Idrees et al., 2021; Kushwaha et al., 2022; Singh et al., 2021).

The approach presented in this paper does not focus on facing security challenges on Blockchain, instead, it leverages this technology to equip the network with a secure public ledger able to support some functionalities. In particular, we exploit Blockchain and Smart Contracts to keep trace of *(i)* the information necessary to discover the identity of aggregators for target nodes; *(ii)* the trust scores assigned by workers to estimate the reliability of an aggregator; and *(iii)* the identity of corrupted objects resulting from the monitoring activity of workers towards target nodes.

Therefore, also because our proposal does not aim at extending existing Blockchain solutions, we do not consider vulnerabilities and possible direct attacks to it. In other words, for Assumption A.4, we presuppose that the underlying Blockchain solution guarantees the standard security requirements already adopted for common Blockchain applications (Singh et al., 2021), thus it can be considered secure.

4.2.4 SP.4 - Resistance to attacks on the Reputation Model

Our strategy includes also a contribution to the computation of a trust score to evaluate the trustworthiness of IoT nodes. Anyway, although in our approach we described a simple adaptation of an existing trust model (Corradini et al., 2022) into our scenario, this task can be considered orthogonal to our strategy. Therefore, for our security analysis related to the trust model we can rely on the analysis conducted in (Corradini et al., 2022).

Anyway, just to give a few examples of attacks targeting the trust model of our approach, we consider in the following how our schema proves to be robust against two of the most popular attacks on reputation systems, namely the White-washing and Slandering (or Bad-mouthing) attack.

The former occurs when a malicious node tries to exit and rejoin the network to delude the system and clean its reliability.

Our strategy is based on a community-oriented general perspective of the trustworthiness of a target node. Indeed, to assess the reliability of a node, we adopt a window-based strategy leveraging our behavioral fingerprinting models. Specifically, trust scores are computed based on the rate of mispredicted symbols inside an observation window. At this point, if the reputation of the node, computed by aggregating all the trust contributions towards it, goes under a reference threshold, it will be isolated by the other peers and, therefore, as explained above it is technically banned from the system (for, at least, a time ϕ_{ban}). Moreover, as an additional security mechanism, if a device is banned multiple times, ϕ_{ban} can be incremented at every ban until the object removal is permanent.

Observe that, in IoT, one of the main issues is related to the difficulty of mapping a unique identifier with an object. Therefore, in some cases, an attacker could still perform a Whitewashing attack by exiting the system and re-introducing his/her device with a different (forged) identifier. To face this situation, we can adopt a pessimistic attitude approach, which imposes that newly introduced devices will start in a banned state (no other node will interact with it) for a time ϕ_{ban}, and only after this period they can be part of the network. In this way, attempting a whitewashing by forging a new identifier for a device would result again in the node being banned for ϕ_{ban} time, and no advantage is obtained.

As for Slandering or Bad-mouthing attacks, they occur when an intruder tries to distort the innocent nodes' reputation by attesting a negative opinion of them. In our approach, a Slandering or Bad-mouthing attack can happen if a worker lies about the result of the behavioral fingerprinting model of a monitored node computing a false negative trust score for that node.

If this threat is performed by a single node, only its local contribution to the trust score is impacted. Hence, the global trust score will not be compromised because it will be balanced by the honest contributions of the other nodes testing the behavioral fingerprinting model for the victim.

Moreover, these attacks can be performed also in a distributed fashion, through some colluding nodes trying to poison the trust score of a victim with multiple negative trust contributions. Anyway, for Assumption A.2, an attacker cannot control the majority of workers holding a behavioral fingerprinting model for a target. It is worth noting that this assumption is commonly accepted for distributed domain scenarios, in which the majority of users or nodes in a network or a system can be considered honest at any time (Cramer et al., 1997; Rottondi et al., 2016; Zwierko et al., 2007). As an additional consideration, our approach preserves the privacy of the identity of the nodes forming the group of workers for an object thanks to HE. Hence, the components of the group do not know each other, also an attacker cannot have this information from additional knowledge derived from any direct physical access to IoT objects for Assumption A.3. For all these reasons, our approach can be considered robust against Slandering or Bad-mouthing attacks.

4.2.5 SP.5 - Resistance to attacks on the IoT network

As for attacks undermining network and node availability, we consider the two most popular ones, namely DoS and Sleep Deprivation attacks.

During a Denial of Service (DoS) an attacker introduces a large amount of (dummy) transactions in the network to overflow it and affect its availability. In our approach, this attack could also result in the impossibility for nodes to run the FL algorithm and check peers' behavior. For this reason, any existing solution aiming at preventing DoS attacks in IoT could be exploited in our approach, such as the ones presented in (Abughazaleh et al., 2020; Baig et al., 2020; Hussain et al., 2020). It is worth explaining that, however, our approach does not add any advantage to an adversary performing such a category of attacks.

A form of DoS attack specific to the IoT environment is known as Sleep Deprivation Attack (SDA, hereafter) whose objective is to undermine the power of the node to consume its battery life and power it off (excluding the victim from the network). As for this attack, our approach natively supports a countermeasure. Indeed, the alteration in the behavior of an attacked node can be detectable by our behavioral fingerprinting models. Therefore, our approach can prevent SDA, because once a change in the behavior of the attacked node is detected, the other nodes can safely discard all the requests coming from it.

5 Experiments

This section deals with the analysis of our experimental campaign useful for validating our approach. In particular, in the next subsections, after the description of our dataset, we report in detail the performance evaluation of our solution to build a global behavioral fingerprinting model using FL, the results of our solution for anomaly detection, and, finally, the tests to assess the performance of the overall approach in terms of execution times.

5.1 The Dataset

To validate our proposal, we started from a dataset publicly available online concerning IoT traffic collected by a centralized network hub. The dataset is available at https://iotanalytics.unsw.edu.au/attack-data.html and has been originally produced by the authors of (Hamza et al., 2019). It contains about 65 GB of data describing daily IoT traffic (i.e., traffic generated by smart devices, such as light sensors, motion sensors, and so forth). The original dataset contains both data generated in the absence of cyber attacks, as well as traffic generated when some attack is deployed on the IoT nodes. Interestingly, this same dataset has been adopted in (Aramini et al., 2022) to test the performance of the original behavioral fingerprinting model which is extended in this proposal. The authors of (Aramini et al., 2022) also enhanced this dataset to simulate the collection of traffic from the IoT nodes, directly (no central hub collector); thus, granting that payload data is accessible from monitoring nodes. Because in our scenario, we are also focusing on a fully distributed context, we adopt the extended version of the above dataset generated in (Aramini et al., 2022). Some statistics about our referring data are, then, reported in Table 3.

Table 3 Statistics of the dataset considered in our study

Communication Type	Min # of packets	Max # of packets
Benign	12,793	97,256
Benign with payload	4,670	39,000
Malign	6,971	89,148
Malign with payload	2,196	8,694

5.2 Performance Analysis of our Global Behavioral Fingerprinting Model

To assess the performance of our approach to build a global behavioral fingerprinting model using FL, we performed a comparison analysis between our solution and the baseline approach proposed in (Aramini et al., 2022). Indeed, the approach of (Aramini et al., 2022) started from the results reported in (Nguyen et al., 2019) and demonstrated that, by exploiting additional features related to the payload, it is possible to improve the solution performance. Indeed, the authors of (Aramini et al., 2022) proposed a fully distributed behavioral fingerprinting model, which, however, is focused on just a point-to-point vision of a node towards a target peer. Our approach, instead, extends this idea by considering that in IoT a node can participate in multiple services, thus showing different behavioral patterns according to them. Therefore, we aim to build a global model considering all such patterns to represent the complete behavior of a target node, and we leverage Federated Learning for this objective.

With that said, we start our comparison by analyzing the performance of our model and the model of (Aramini et al., 2022) for 12 nodes monitoring 3 different targets. As for our approach, we extracted from the original dataset groups of nodes having communications with the same targets; in this way, we could build our Federated Learning scenario. In particular, after analyzing all the communications available in the dataset, we were able to set the number of workers to 4. Hence, for each target, we obtained a global model built according to our strategy and 4 point-to-point models built according to the strategy of (Aramini et al., 2022). As for the training data, we used the communication sequences but we kept the 20% of them for the subsequent testing. Indeed, once the models have been built, to compare the obtained performance, we used the test set of each involved node, independently. Of course, the point-to-point (P2P) models are trained and tested on the data of the same communication (direct testing), whereas our global model (GM) is trained on global data and, then, tested on the individual test sets of the involved nodes; thus, we can expect a slight reduction in the performance. However, we argue that such a reduction is negligible. The results of this experiment are reported in Table 4 where we analyzed prediction accuracy results and in which c_1, c_2, c_3, and c_4, for each target node, act as both individual nodes building $P2P$ models of the target behavior and the workers of the Federated Learning task building the global model GM.

Table 4 Comparison of the performance of our approach (GM) and the solution of (Aramini et al., 2022) (P2P) with direct testing in terms of prediction accuracy

	Model	c_1	c_2	c_3	c_4
Target 1	P2P	**0.78**	0.75	**0.86**	**0.83**
	GM	0.77	**0.76**	0.82	**0.83**
Target 2	P2P	0.81	**0.82**	**0.85**	**0.83**
	GM	**0.82**	0.80	0.75	**0.83**
Target 3	P2P	0.82	**0.89**	0.74	**0.84**
	GM	**0.86**	**0.89**	**0.79**	**0.84**

By analyzing this table we can see that, as expected, the point-to-point models achieve sometimes slightly better performance when tested against a test set derived by the same communication from which the training set has been extracted. However, our hypothesis is also correct as the performance reduction of our approach is negligible (less than 1%, on average).

However, the characteristic of our global model is just the capability of being generally valid for any communication towards a target node (also for communications related to different services). To test this aspect, we proceeded with a similar experiment as above, but we performed a cross-testing and assessed the performance of each point-to-point model ($P2P_{c_1}$, $P2P_{c_2}$, $P2P_{c_3}$, and $P2P_{c_4}$) and our global one, on every test set available from the different involved nodes. We reported the results of this experiment in Table 5.

In practice, in our testbed, each client owns a dataset referring to its individual communications with the shared target node. From these datasets, for each client, we extracted a test set namely, Test-set c_1, Test-set c_2, Test-set c_3, and Test-set c_4, respectively. At this point, differently from the previous experiment, the cross-testing consisted in applying all the P2P models and our global one on all the available test sets from the clients. Of course, when a P2P model, say $P2P_{c_1}$, is applied to the test set belonging to the client that built

Table 5 Comparison of the performance of our approach and the solution of (Aramini et al., 2022) with cross testing

#Model	Test-set c_1	Test-set c_2	Test-set c_3	Test-set c_4
$P2P_{c_1}$	**0.82**	< 0.01	< 0.01	< 0.01
$P2P_{c_2}$	< 0.01	**0.89**	< 0.01	< 0.01
$P2P_{c_3}$	< 0.01	< 0.01	**0.74**	< 0.01
$P2P_{c_4}$	< 0.01	< 0.01	< 0.01	**0.84**
GM	**0.86**	**0.89**	**0.79**	**0.84**

this model, c_1 in this case, the experiment implies a direct testing, thus returning the optimal performance for that specific model. With this experiment, we aim at demonstrating that, because the communications of different clients with the same target node may concern different services, local P2P models are not a general solution to monitor the behavior of a node.

As a matter of fact, by inspecting Table 5, we can clearly see that the point-to-point models return satisfactory accuracy results only when applied to the test set generated by the same communication of the original training set (direct testing). The last row of this table, instead, shows the performance of our global model which is very satisfactory across every considered test set. This confirms our intuition that classical behavioral fingerprinting approaches, such as (Aramini et al., 2022) and (Nguyen et al., 2019), reach very satisfactory performance assessing the behavior of a node concerning *only* a single target communication type (i.e., communications generated for a specific service or action). Our approach, on the other hand, allows for the construction of consistent and complete behavioral fingerprints of an IoT node. In practice, the models built by our approach are more stable and can be used to characterize the behavior of a target node in general, and not just for a specific single service/action it may offer/perform.

5.3 Windows-Based Anomaly Detection with Behavioral Fingerprint

As described in Section 3.3.2, our approach exploits behavioral fingerprinting models to detect anomalies on target nodes by leveraging a window-based mechanism. In particular, once again, our solution is based on the strategy originally described (Nguyen et al., 2019) and (Aramini et al., 2022).

The proposed strategy works by computing the misprediction rate of the next symbol inside an observation window. As seen in Section 3.4, the misprediction rate is defined as the ratio between the number of symbols inside the windows not predicted by our behavioral fingerprinting model as plausible ones in the analyzed sequence and the overall number of symbols in the observation window. Clearly, the choice of the right size for such a window plays a key role. Intuitively, larger windows imply a more stable anomaly detection capability, as any noise, even the one caused by the errors in the predictions introduced by our model, would be smoothed out (smaller oscillations in the misprediction curve). Of course, the larger the window the slower the detection of possible anomalies, since more symbols (and, hence, more packets) would be required to detect it. A possible, strategy for identifying the correct size is to use the difference between the maximum and minimum peaks of the misprediction curve. Indeed, a lower difference would imply better stability. At this point, to find the optimal solution we can rely on the *Kneedle* algorithm (Satopaa et al., 2011). Specifically, it seeks to find the elbow/knee in the misprediction curve, which corresponds to the point where the curve has the most visible change from high slope to low slope. In Fig. 5, we show the application of this algorithm in our context.

As shown in this figure, in our scenario, a possible optimal configuration for the window is 100 symbols.

With this setting, we performed a further experiment to demonstrate the capability of our solution to detect anomalies in the behavior of an IoT node and we compared the obtained performance with those obtained by related point-to-point models. Specifically, we focused again on the testbed introduced in the experiment described in Section 5.2, in which we considered 4 different point-to-point behavioral fingerprinting models (P2P models, for short), according to the strategy of (Aramini et al., 2022), built by 4 IoT nodes, namely c_1, c_2, c_3, and c_4, and targeting the same node b. Moreover, we simulated an FL task involving the same 4 nodes and built a global model for b (GM, for short) according to our approach. Of course, each involved monitoring node, $c_1, \ldots, c_4$, collects the portion of traffic originated by b towards it and creates its training and test sets. At this point, we analyzed the performance of the window-based anomaly detection strategy using both the P2P models and the GM model as underlying fingerprinting models. To do so, we fixed a threshold of 0.5 (i.e., 50% of the symbols in a window), so that a misprediction rate higher than this threshold in a window would correspond to the detection of an anomalous behavior. Moreover, we simulated the situation in which the first 280 packets from b are benign and after that, the node performs an attack. To simulate the attack, we

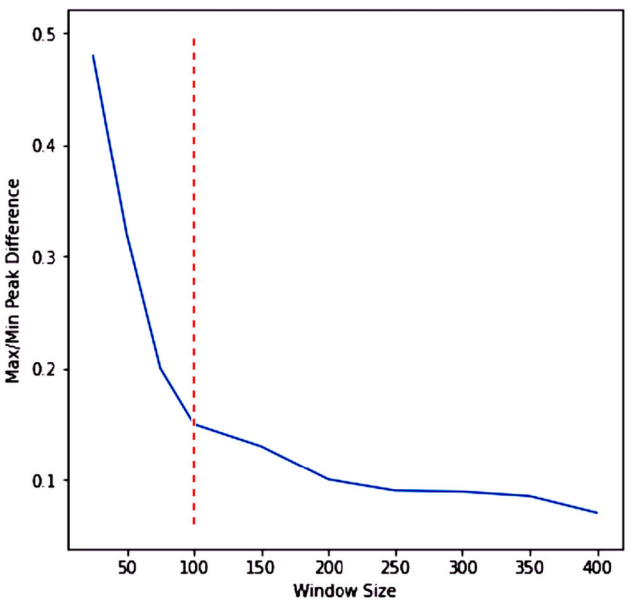

Fig. 5 Application of the Kneedle algorithm to identify the best window size

used the malign traffic for this node contained in our original dataset (see Section 5.1). The obtained results are visible in Fig. 6.

As shown in this figure, the anomaly detection strategy using P2P models works only when the traffic analyzed is derived from the test set of the node that built the underlying P2P model. Instead, when it is applied to different test sets it cannot distinguish between normal and anomalous behaviors. When our GM model is used instead, the anomaly detection strategy achieves very good performance across all the different test sets (see the subplots in the last line of Fig. 6). This allows for the construction of a solid anomaly detection solution for IoT nodes, which is agnostic on the specific services the monitored nodes could be involved in.

5.4 Analysis of Execution Times

This section is devoted to the tests performed to validate the feasibility and effectiveness of our proposal in terms of execution times. Indeed, our approach is designed for an IoT scenario, typically characterized by many heterogeneous devices.

We start by considering our privacy-preserving schema for the identification of the correct aggregator of a node (Algo-

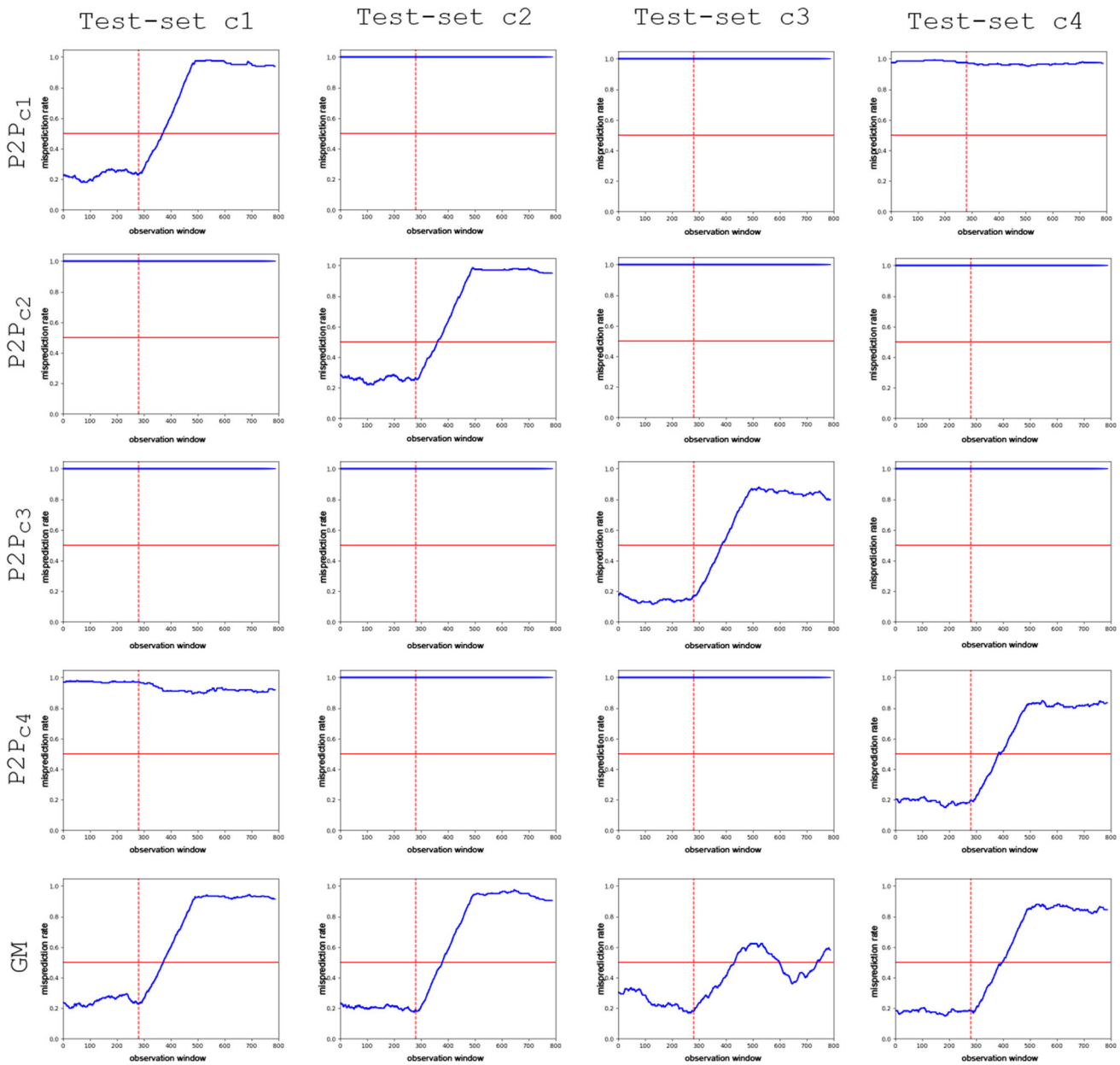

Fig. 6 Performance of the window-based anomaly detection strategy using both P2P and GM models to monitor a common target

Table 6 Average execution times of Algorithm 1 on different device types

Device Type	Average MPC Time
Desktop PC	49.6 ms
Raspberry Pi4	185.3 ms
1 core ARM1176 (QEMU)	774 ms

rithm 1), and for the creation of groups of workers for the FL tasks (Algorithm 2). Both cases share a similar strategy and are based on the computation of bitwise XOR operations on hashed value through homomorphic hashing. Therefore, we focus here on Algorithm 1, which is based on Equation 1, and, hence, test the feasibility of this computation on different types of devices. For this experiment, we considered the same Federated Learning scenario analyzed in the previous experiment and derived from the original dataset. Moreover, we considered 3 types of device, namely: *(i)* a desktop personal computer equipped with a Ryzen 7 5800x Octa-core 3.8 GHz base, 4.7 GHz boost processor, and 32GB of RAM, *(ii)* a Raspberry Pi4 with a Quad-core Cortex-A72 processor and 8GB of RAM, and *(iii)* a single-core ARM1176 CPU with 512MB of RAM, emulated with the QEMU virtualization environment[2]. We executed Algorithm 1 on each considered device type and reported the results in Table 6.

By inspecting this table, we can conclude that our privacy-preserving scheme is feasible for all the considered device types. The computation is, in general, carried out in less than 1 second with a maximum value of 774 milliseconds for the less capable considered device type.

After that, we focused on the computational requirements for the aggregator in our solution. Aggregators coordinate Federated Learning tasks and, during each training epoch, aggregate the gradient updates produced by the workers to build the global model.

To evaluate the execution times of the aggregation task, we considered, again, the 3 types of device and the Federated Learning task mentioned above. Hence, we measured the time required, on average, to aggregate the gradient updates of the local models (i.e., of the local GRU deep learning models described in Section 3.3) during the epochs of such a Federated Learning task. The result of this experiment is reported in Table 7.

This result confirms again that both our secure multi-party computation and the aggregation task can be executed by very heterogeneous devices including those with limited computational capability (such as a node equipped with a single core ARM1176 and 512MB of RAM).

As a final evaluation of execution times, we focused on the performance of the inference of a trained instance of our behavioral fingerprinting model. In particular, we analyzed the impact of our secure delegation strategy in such a task to validate its feasibility. Therefore, we executed model inferences with and without the secure delegation strategy and computed the execution times for batches of consecutive symbols of variable sizes. The obtained results are reported in Fig. 7.

Table 7 Average aggregation time for different device types

Device Type	Average Aggregation Time
Desktop PC	118ms
Raspberry Pi4	241ms
1 core ARM1176 (QEMU)	755ms

This figure shows that the performance reduction introduced by our secure delegation strategy is about 16.6% on average. Although such a difference is not negligible, the very low general inference times of our model make the inclusion of the delegation strategy still feasible across all the possible scenarios.

6 Discussion and Conclusion

In recent years, IoT devices have grown in number and complexity to empower new applications with enhanced possibilities in monitoring, decision-making, and automation contexts. Clearly, in this scenario, privacy and security aspects become a major concern.

This paper provides a contribution to this setting by designing a novel distributed framework for the computation

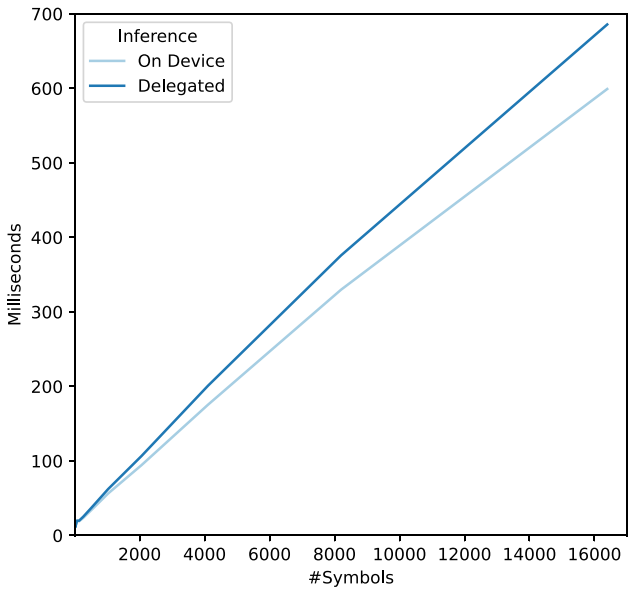

Fig. 7 Inference time with and without our secure delegation strategy

[2] https://www.qemu.org/

of *global* behavioral fingerprints of objects. Indeed, classical behavioral fingerprints are based on Machine Learning solutions to model object interactions and assess the correctness of their actions. Still, scalability, privacy, and intrinsic limitations of adopted Machine Learning algorithms represent the main aspects to be improved to make this paradigm entirely suitable for the IoT environment. Indeed, in classical distributed fingerprinting approaches, an object models the behavior of a target contact by exploiting only the information coming from the direct interaction with it, which represents a very limited view of the target because it does not consider services and messages exchanged with other neighbors. However, building global models with information coming from several interactions of nodes with the target may lead to critical privacy concerns.

To face this issue, we assumed a comprehensive perspective analyzing the hidden patterns of the behavior of a node in the interactions with all its peers over a network. To do so, we designed a solution based on Federated Learning that benefits from a distributed computation of behavioral fingerprints involving different working nodes. Thanks to this novel ML strategy, besides enriching the fingerprinting model with information coming from different interactions of multiple nodes, our approach addresses also several aspects related to the security and privacy of data exchanged among the involved actors. Moreover, it guarantees the scalability of the proposed solution and very satisfactory accuracy results of the anomaly detection schema making our approach suitable to the constantly changing attack surface that characterizes the modern IoT. Furthermore, our solution considers the intrinsic heterogeneity of the entities involved in the considered scenario, allowing less capable nodes to participate in the framework, by relying on a secure delegation strategy for both the training and the inference of FL models in a privacy-preserving way. Finally, through the properties of Homomorphic Encryption and the Blockchain technology, our approach guarantees the privacy of both the target object and the different contributors, as well as the robustness of the solution in the presence of security attacks. All these features lead to a secure fully privacy-preserving solution whose robustness and correctness have been evaluated in this paper through a detailed security analysis. Moreover, an extensive experimental campaign showed that the performance of our model is very satisfactory, and we can distinguish between normal and anomalous behavior across every considered test set, reaching a 0.85 value of accuracy on average. Furthermore, the very low general inference times of our model make the inclusion of the delegation strategy still feasible across all the possible scenarios with a performance reduction of only 16.6%, on average.

While this work has provided valuable insights into the potential of our solution for anomaly detection in IoT, several limitations should be acknowledged. Firstly, our framework needs a sufficient total number of heterogeneous nodes to perform its operations properly. Moreover, even if secure delegation can be applied, still an adequate number of powerful nodes with sufficient computational capability, memory, and stability should be present to train local ML models. Furthermore, the effectiveness of our approach, which is based on FL, heavily relies on frequent communications between the aggregator and the workers in the training phase. In an IoT scenario, this might lead to longer training times and potentially hinder convergence. Anyway, a number of recent studies have already tackled the issue of training distributed machine learning models for resource-constrained IoT devices (Imteaj et al., 2021). Our work can leverage one of the existing studies on the application of FL to IoT since this part is orthogonal to our work.

We plan to expand the research described in this proposal with further investigations in the next future. For instance, we are planning to study a solution to build, still in a collaborative and distributed way, the behavioral fingerprinting of objects in the network but also taking into account an optimized orchestration of their workload. In particular, thanks to secure delegation, this solution should allow a better distribution of the workload, generated by FL tasks, among the nodes of the network, according to power consumption minimization, Service Level Agreement (SLA, for short) requirements, and the reliability of the nodes.

Acknowledgements This work was supported in part by the project SERICS (PE00000014) under the NRRP MUR program funded by the EU-NGEU, and by the Italian Ministry of University and Research through the PRIN Project "HOMEY: a Human-centric IOE-based framework for supporting the transition towards industry 5.0" (code 2022NX7WKE).

Funding Open access funding provided by Università degli Studi di Pavia within the CRUI-CARE Agreement.

Availability of data and materials The dataset used in this paper is publicly available in the repository: https://iotanalytics.unsw.edu.au/attack-data.html and has been originally produced by the authors of (Hamza et al., 2019). In this paper, we also adopted the algorithms proposed in Aramini et al. (2022) to generate payload data.

Declarations

Conflict of interest/Competing interests The authors declare that they have no conflict of interest or competing interests that are relevant to the content of this article.

References

Abughazaleh, N., Bin, R., & Btish, M. (2020). Dos attacks in iot systems and proposed solutions. *Int. J. Comput. Appl., 176*(33), 16–19.

Adat, V., & Gupta, B. B. (2018). Security in internet of things: issues, challenges, taxonomy, and architecture. *Telecommunication Systems, 67*(3), 423–441.

Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I., & Guizani, M. (2020). A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Communications Surveys & Tutorials, 22*(3), 1646–1685.

Ali, M., Karimipour, H., & Tariq, M. (2021). Integration of blockchain and federated learning for internet of things: Recent advances and future challenges. *Computers & Security, 108*, 102355.

Al-Sarawi, S., Anbar, M., Abdullah, R., Al Hawari, A.B. (2020). in *2020 Fourth World Conference on smart trends in systems, security and sustainability (WorldS4)* (IEEE), pp. 449–453

Aramini, A., Arazzi, M., Facchinetti, T., Ngankem, L.S., Nocera, A. (2022). in *2022 IEEE 18th International Conference on Factory Communication Systems (WFCS)* (IEEE), pp. 1–8

Baig, Z. A., Sanguanpong, S., Firdous, S. N., Nguyen, T. G., & So-In, C. (2020). Averaged dependence estimators for dos attack detection in iot networks. *Future Generation Computer Systems, 102*, 198–209.

Bellare, M., Goldreich, O., Goldwasser, S. (1994). in *Annual International Cryptology Conference* (Springer), pp. 216–233

Bezawada, B., Bachani, M., Peterson, J., Shirazi, H., Ray, I., Ray I. (2018). in *Proceedings of the 2018 workshop on attacks and solutions in hardware security*, pp. 41–50

Blanco-Justicia, A., Domingo-Ferrer, J., Martínez, S., Sánchez, D., Flanagan, A., & Tan, K. E. (2021). Achieving security and privacy in federated learning systems: Survey, research challenges and future directions. *Engineering Applications of Artificial Intelligence, 106*, 104468.

Buccafurri, F., Lax, G., Nicolazzo, S., & Nocera, A. (2016). A privacy-preserving localization service for assisted living facilities. *IEEE Transactions on Services Computing, 13*(1), 16–29.

Cauteruccio, F., Fortino, G., Guerrieri, A., Liotta, A., Mocanu, D. C., Perra, C., Terracina, G., & Vega, M. T. (2019). Short-long term anomaly detection in wireless sensor networks based on machine learning and multi-parameterized edit distance. *Information Fusion, 52*, 13–30.

Celdrán, A.H., Sánchez, P.M.S., Castillo, M.A., Bovet, G., Pérez, G.M., Stiller, B. (2022). Intelligent and behavioral-based detection of malware in iot spectrum sensors. *International Journal of Information Security* pp. 1–21

Chen, Y., Lu, Y., Bulysheva, L., Kataev, M.Y. (2022). Applications of blockchain in industry 4.0: A review. *Information Systems Frontiers* pp. 1–15

Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *Ieee Access, 4*, 2292–2303.

Corradini, E., Nicolazzo, S., Nocera, A., Ursino, D., & Virgili, L. (2022). A two-tier Blockchain framework to increase protection and autonomy of smart objects in the IoT. *Computer Communications, 181*, 338–356.

Cramer, R., Gennaro, R., & Schoenmakers, B. (1997). A secure and optimally efficient multi-authority election scheme. *European transactions on Telecommunications, 8*(5), 481–490.

Dedeoglu, V., Jurdak, R., Putra, G. D., Dorri, A., Kanhere, S. S. (2019). in *Proceedings of the 16th EAI international conference on mobile and ubiquitous systems: computing, networking and services*, pp. 190–199

Ferretti, M., Nicolazzo, S., & Nocera, A. (2021). H2O: Secure Interactions in IoT via Behavioral Fingerprinting. *Future Internet, 13*(5), 117.

Gentry, C. (2009). *A fully homomorphic encryption scheme* (Stanford university)

Hamad, S.A., Zhang, W.E., Sheng, Q.Z., Nepal, S. (2019). in *2019 18th IEEE international conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering (TrustCom/BigDataSE)* (IEEE), pp. 103–111

Hammi, M. T., Hammi, B., Bellot, P., & Serhrouchni, A. (2018). Bubbles of trust: A decentralized blockchain-based authentication system for iot. *Computers & Security, 78*, 126–142.

Hamza, A., Gharakheili, H. H., Benson, T. A., Sivaraman, V. (2019). in *Proceedings of the 2019 ACM Symposium on SDN Research*, pp. 36–48

Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P., & Sikdar, B. (2019). A survey on iot security: application areas, security threats, and solution architectures. *IEEE Access, 7*, 82721–82743.

Hussain, F., Abbas, S. G., Husnain, M., Fayyaz, U. U., Shahzad, F., Shah, G. A. (2020). in *2020 IEEE 23rd International Multitopic Conference (INMIC)* (IEEE), pp. 1–6

Idrees, S. M., Nowostawski, M., Jameel, R., & Mourya, A. K. (2021). Security aspects of blockchain technology intended for industrial applications. *Electronics, 10*(8), 951.

Imteaj, A., Thakker, U., Wang, S., Li, J., & Amini, M. H. (2021). A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal, 9*(1), 1–24.

Jere, M. S., Farnan, T., & Koushanfar, F. (2020). A taxonomy of attacks on federated learning. *IEEE Security & Privacy, 19*(2), 20–28.

Khalil, U., Ahmad, A., Abdel-Aty, A. H., Elhoseny, M., El-Soud, M. W. A., & Zeshan, F. (2021). Identification of trusted iot devices for secure delegation. *Computers & Electrical Engineering, 90*, 106988.

Khan, L. U., Saad, W., Han, Z., Hossain, E., Hong, C. S. (2021). Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*

Khan, M. A., & Salah, K. (2018). Iot security: Review, blockchain solutions, and open challenges. *Future generation computer systems, 82*, 395–411.

Kim, Y. S., & Heo, J. (2012). Device authentication protocol for smart grid systems using homomorphic hash. *Journal of Communications and Networks, 14*(6), 606–613.

Kim, M., Song, Y., Wang, S., Xia, Y., & Jiang, X. (2018). Secure logistic regression based on homomorphic encryption: Design and evaluation. *JMIR medical informatics, 6*(2), e8805.

Kohno, T., Broido, A., & Claffy, K. C. (2005). Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing, 2*(2), 93–108.

Konečnỳ, J., McMahan, B., Ramage, D. (2015). Federated optimization: Distributed optimization beyond the datacenter. arXiv preprint arXiv:1511.03575

Kozlov, D., Veijalainen, J., Ali, Y. (2012). in *BODYNETS*, pp. 256–262

Kushwaha, S. S., Joshi, S., Singh, D., Kaur, M., Lee, H. N. (2022). Systematic review of security vulnerabilities in ethereum blockchain smart contract. *IEEE Access*

Lewi, K., Kim, W., Maykov, I., Weis, S. (2019). Securing update propagation with homomorphic hashing. *Cryptology ePrint Archive*

Li, S., Xu, L. D., & Zhao, S. (2015). The internet of things: a survey. *Information systems frontiers, 17*, 243–259.

Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.R., Tarkoma, S. (2017). in *2017 IEEE 37th International Con-*

ference on Distributed Computing Systems (ICDCS) (IEEE), pp. 2177–2184
Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., Sadeghi, A. R. (2019). in *2019 IEEE 39th International conference on distributed computing systems (ICDCS)* (IEEE), pp. 756–767
Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J., & Poor, H. V. (2021). Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials, 23*(3), 1622–1658.
Nofer, M., Gomber, P., Hinz, O., & Schiereck, D. (2017). Blockchain. Business & Information. *Systems Engineering, 59*(3), 183–187.
Oser, P., Kargl, F., Lüders, S. (2018) in *International conference on security, privacy and anonymity in computation, communication and storage* (Springer), pp. 417–427
Peralta, G., Cid-Fuentes, R. G., Bilbao, J., & Crespo, P. M. (2019). Homomorphic encryption and network coding in iot architectures: Advantages and future challenges. *Electronics, 8*(8), 827.
Pietro, R. D., Salleras, X., Signorini, M., Waisbard, E. (2018). in *Proc. of the ACM International Symposium on Access Control Models and Technologies (SACMAT'18)* (Indianapolis, IN, USA), pp. 77–83. ACM
Preuveneers, D., Rimmer, V., Tsingenopoulos, I., Spooren, J., Joosen, W., & Ilie-Zudor, E. (2018). Chained anomaly detection models for federated learning: An intrusion detection case study. *Applied Sciences, 8*(12), 2663.
Radhakrishnan, S. V., Uluagac, A. S., & Beyah, R. (2014). Gtid: A technique for physical device and device type fingerprinting. *IEEE Transactions on Dependable and Secure Computing, 12*(5), 519–532.
Rana, M., Mamun, Q., & Islam, R. (2022). Lightweight cryptography in iot networks: A survey. *Future Generation Computer Systems, 129*, 77–89.
Ren, W., Tong, X., Du, J., Wang, N., Li, S. C., Min, G., Zhao, Z., & Bashir, A. K. (2021). Privacy-preserving using homomorphic encryption in mobile iot systems. *Computer Communications, 165*, 105–111.
Rey, V., Sánchez, P. M. S., Celdrán, A. H., & Bovet, G. (2022). Federated learning for malware detection in iot devices. *Computer Networks, 204*, 108693.
Rottondi, C., Panzeri, A., Yagne, C. T., & Verticale, G. (2016). Detection and mitigation of the eclipse attack in chord overlays. *International Journal of Computational Science and Engineering, 13*(2), 111–121.
Sánchez, P. M. S., Celdrán, A. H., Rubio, J. R. B., Bovet, G., Pérez, G. M. (2021). Robust federated learning for execution time-based device model identification under label-flipping attack. arXiv preprint arXiv:2111.14434
Sánchez, P. M. S., Celdrán, A. H., Schenk, T., Iten, A.L.B., Bovet, G., Pérez, G. M., Stiller, B. (2022). Studying the robustness of anti-adversarial federated learning models detecting cyberattacks in iot spectrum sensors. arXiv preprint arXiv:2202.00137
Sánchez, P. M. S., Valero, J. M. J., Celdrán, A. H., Bovet, G., Pérez, M. G., & Pérez, G. M. (2021). A survey on device behavior fingerprinting: Data sources, techniques, application scenarios, and datasets. *IEEE Communications Surveys & Tutorials, 23*(2), 1048–1077.
Satopaa, V., Albrecht, J., Irwin, D., Raghavan, B. (2011). in *2011 31st international conference on distributed computing systems workshops* (IEEE), pp. 166–171
Shafagh, H., Hithnawi, A., Burkhalter, L., Fischli, P., Duquennoy, S. (2017). in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, pp. 1–14
Shrestha, R., Kim, S. (2019). in *Advances in Computers*, vol. 115 (Elsevier), pp. 293–331
Sicari, S., Cappiello, C., De Pellegrini, F., Miorandi, D., & Coen-Porisini, A. (2016). A security-and quality-aware system architecture for internet of things. *Information Systems Frontiers, 18*, 665–677.
Singh, S., Hosen, A. S., & Yoon, B. (2021). Blockchain security attacks, challenges, and solutions for the future distributed iot network. *IEEE Access, 9*, 13938–13959.
Tweneboah-Koduah, S., Skouby, K. E., & Tadayoni, R. (2017). Cyber security threats to iot applications and service domains. *Wireless Personal Communications, 95*, 169–185.
Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., & Yu, H. (2019). Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning, 13*(3), 1–207.
Yao, H., Wang, C., Hai, B., Zhu, S. (2018). in *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)* (IEEE), pp. 243–248
Zwierko, A., & Kotulski, Z. (2007). A light-weight e-voting system with distributed trust. *Electronic Notes in Theoretical Computer Science, 168*, 109–126.

Marco Arazzi is currently a Ph.D. Student in Computer Engineering at the same University. From March to July 2023, he worked as a Visiting Researcher in the Cyber Security group of the Delft University of Technology (TU Delft). His research interests include Data Science, Machine Learning, Social Network Analysis, the Internet of Things, Privacy, and Security. He is the author of 10 scientific papers in these research fields.

Serena Nicolazzo is currently a Type-A Temporary Research Fellow (RTDA) at the University of Milan. She got a PhD in Information Engineering at the University Mediterranea of Reggio Calabria in 2017. Her research interests include Data Science, Security, Privacy, and Social Network Analysis. She is involved in several TPCs and editorial boards of prestigious International Conferences and Journal in the context of Data Science and Cybersecurity and she is the author of about 40 scientific papers. She was a Visiting Researcher at Middlesex University of London and is actively collaborating with the Polytechnic University of Marche, the University of Pavia, and the University College of London.

Antonino Nocera is an Associate Professor at the University of Pavia. He received his PhD in Information Engineering at the Mediterranea University of Reggio Calabria in 2013. His research interests span several research contexts including Artificial Intelligence, Data Science, Security, Privacy, Social Network Analysis, Recommender Systems, Internet of Things, Cloud Computing, and Blockchain. In these research fields, he published about 90 scientific papers. He is involved in several TPCs of prestigious International Conferences in the context of Data Science and Cybersecurity and is an Associate Editor of Information Sciences (Elsevier) and of the IEEE Transactions on Information Forensics and Security.

Authors and Affiliations

Marco Arazzi[1] · Serena Nicolazzo[2] · Antonino Nocera[1]

Marco Arazzi
marco.arazzi01@universitadipavia.it

Serena Nicolazzo
serena.nicolazzo@unimi.it

[1] Department of Electrical, Computer and Biomedical Engineering, University of Pavia, Via A. Ferrata, 5, Pavia 27100, PV, Italy

[2] Department of Computer Science, University of Milan, Via Celoria, 18, Milan 20133, MI, Italy