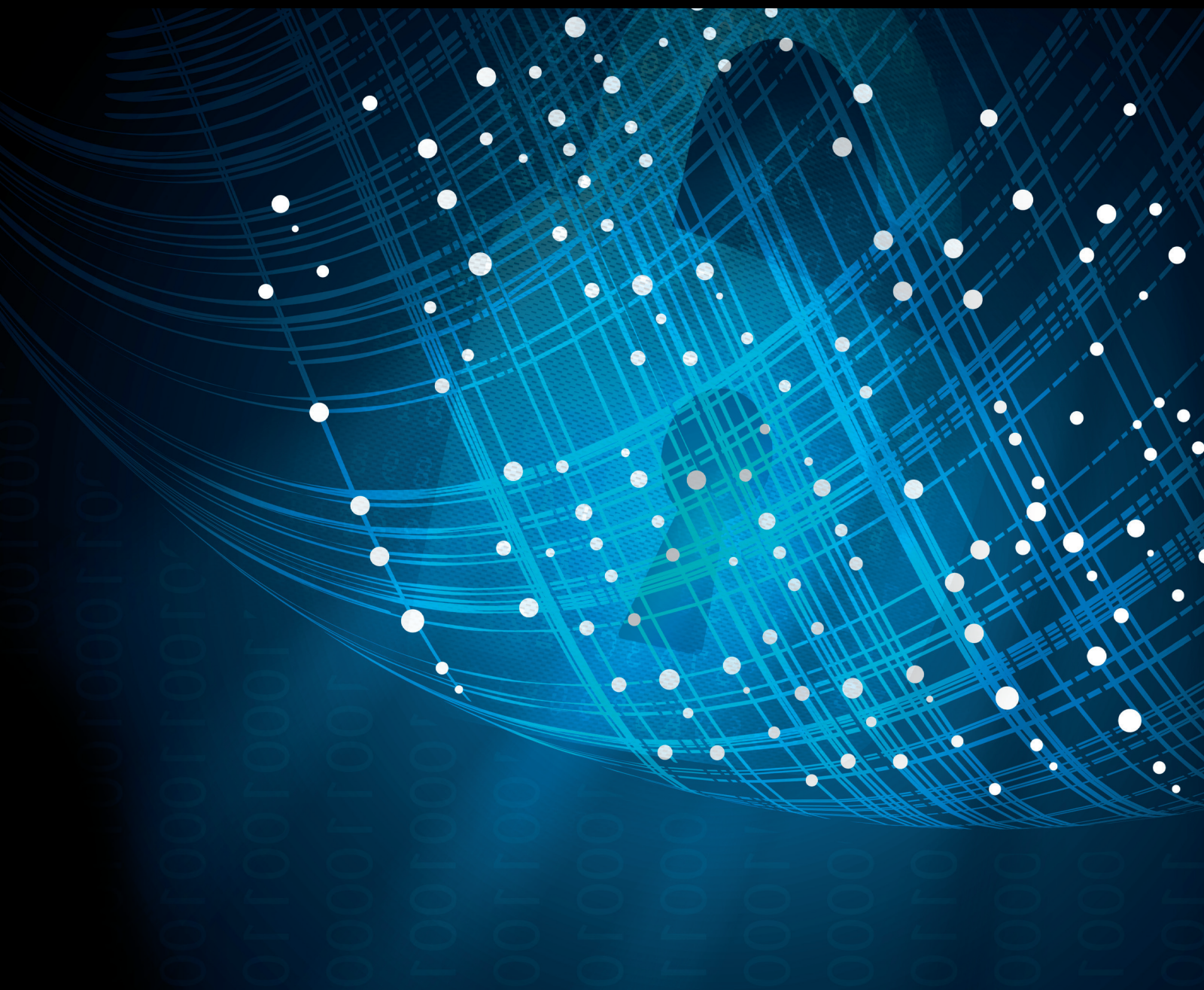


Trustworthy Networking for Beyond 5G Networks 2020

Lead Guest Editor: Zheng Yan

Guest Editors: Raimo Kantola, Qinghua Zheng, Liming Fang, and Aniello Castiglione





**Trustworthy Networking for Beyond 5G
Networks 2020**

Security and Communication Networks

Trustworthy Networking for Beyond 5G Networks 2020

Lead Guest Editor: Zheng Yan

Guest Editors: Raimo Kantola, Qinghua Zheng,
Liming Fang, and Aniello Castiglione






Copyright © 2021 Hindawi Limited. All rights reserved.

This is a special issue published in "Security and Communication Networks." All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chief Editor

Roberto Di Pietro, Saudi Arabia

Associate Editors

Jiankun Hu , Australia
Emanuele Maiorana , Italy
David Megias , Spain
Zheng Yan , China

Academic Editors

Saed Saleh Al Rabae , United Arab Emirates
Shadab Alam, Saudi Arabia
Goutham Reddy Alavalapati , USA
Jehad Ali , Republic of Korea
Jehad Ali, Saint Vincent and the Grenadines
Benjamin Aziz , United Kingdom
Taimur Bakhshi , United Kingdom
Spiridon Bakiras , Qatar
Musa Balta, Turkey
Jin Wook Byun , Republic of Korea
Bruno Carpentieri , Italy
Luigi Catuogno , Italy
Ricardo Chaves , Portugal
Chien-Ming Chen , China
Tom Chen , United Kingdom
Stelvio Cimato , Italy
Vincenzo Conti , Italy
Luigi Coppolino , Italy
Salvatore D'Antonio , Italy
Juhriyansyah Dalle, Indonesia
Alfredo De Santis, Italy
Angel M. Del Rey , Spain
Roberto Di Pietro , France
Wenxiu Ding , China
Nicola Dragoni , Denmark
Wei Feng , China
Carmen Fernandez-Gago, Spain
AnMin Fu , China
Clemente Galdi , Italy
Dimitrios Geneiatakis , Italy
Muhammad A. Gondal , Oman
Francesco Gringoli , Italy
Biao Han , China
Jinguang Han , China
Khizar Hayat, Oman
Azeem Irshad, Pakistan

M.A. Jabbar , India
Minho Jo , Republic of Korea
Arijit Karati , Taiwan
ASM Kayes , Australia
Farrukh Aslam Khan , Saudi Arabia
Fazlullah Khan , Pakistan
Kiseon Kim , Republic of Korea
Mehmet Zeki Konyar, Turkey
Sanjeev Kumar, USA
Hyun Kwon, Republic of Korea
Maryline Laurent , France
Jegatha Deborah Lazarus , India
Huaizhi Li , USA
Jiguo Li , China
Xueqin Liang, Finland
Zhe Liu, Canada
Guangchi Liu , USA
Flavio Lombardi , Italy
Yang Lu, China
Vincente Martin, Spain
Weizhi Meng , Denmark
Andrea Michienzi , Italy
Laura Mongioi , Italy
Raul Monroy , Mexico
Naghme Moradpoor , United Kingdom
Leonardo Mostarda , Italy
Mohamed Nassar , Lebanon
Qiang Ni, United Kingdom
Mahmood Niazi , Saudi Arabia
Vincent O. Nyangaresi, Kenya
Lu Ou , China
Hyun-A Park, Republic of Korea
A. Peinado , Spain
Gerardo Pelosi , Italy
Gregorio Martinez Perez , Spain
Pedro Peris-Lopez , Spain
Carla Ràfols, Germany
Francesco Regazzoni, Switzerland
Abdalhossein Rezai , Iran
Helena Rifà-Pous , Spain
Arun Kumar Sangaiah, India
Nadeem Sarwar, Pakistan
Neetesh Saxena, United Kingdom
Savio Sciancalepore , The Netherlands

De Rosal Ignatius Moses Setiadi ,
Indonesia
Wenbo Shi, China
Ghanshyam Singh , South Africa
Vasco Soares, Portugal
Salvatore Sorce , Italy
Abdulhamit Subasi, Saudi Arabia
Zhiyuan Tan , United Kingdom
Keke Tang , China
Je Sen Teh , Australia
Bohui Wang, China
Guojun Wang, China
Jinwei Wang , China
Qichun Wang , China
Hu Xiong , China
Chang Xu , China
Xuehu Yan , China
Anjia Yang , China
Jiachen Yang , China
Yu Yao , China
Yinghui Ye, China
Kuo-Hui Yeh , Taiwan
Yong Yu , China
Xiaohui Yuan , USA
Sherali Zeadally, USA
Leo Y. Zhang, Australia
Tao Zhang, China
Youwen Zhu , China
Zhengyu Zhu , China

Contents

Predicting the APT for Cyber Situation Comprehension in 5G-Enabled IoT Scenarios Based on Differentially Private Federated Learning

Xiang Cheng , Qian Luo, Ye Pan, Zitong Li, Jiale Zhang, and Bing Chen 

Research Article (14 pages), Article ID 8814068, Volume 2021 (2021)

SANS: Self-Sovereign Authentication for Network Slices

Xavier Salleras  and Vanesa Daza 

Research Article (8 pages), Article ID 8823573, Volume 2020 (2020)

Research Article

Predicting the APT for Cyber Situation Comprehension in 5G-Enabled IoT Scenarios Based on Differentially Private Federated Learning

Xiang Cheng ^{1,2}, Qian Luo,¹ Ye Pan,¹ Zitong Li,² Jiale Zhang,² and Bing Chen ²

¹The Second Research Institute of CAAC, Chengdu 610041, China

²College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 21106, China

Correspondence should be addressed to Bing Chen; cb_china@nuaa.edu.cn

Received 17 September 2020; Revised 16 October 2020; Accepted 9 April 2021; Published 22 April 2021

Academic Editor: Luigi Coppolino

Copyright © 2021 Xiang Cheng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Driven by the advancements in 5G-enabled Internet of Things (IoT) technologies, the IoT devices have shown an explosive growth trend with massive data generated at the edge of the network. However, IoT systems exhibit inherent vulnerability for diverse attacks, and Advanced Persistent Threat (APT) is one of the most powerful attack models that could lead to a significant privacy leakage of systems. Moreover, recent detection technologies can hardly meet the demands of effective security defense against APTs. To address the above problems, we propose an APT Prediction Method based on Differentially Private Federated Learning (APTPMFL) to predict the probability of subsequent APT attacks occurring in IoT systems. It is the first time to apply a federated learning mechanism for aggregating suspicious activities in the IoT systems, where the APT prediction phase does not need any correlation rules. Moreover, to achieve privacy-preserving property, we further adopt a differentially private data perturbation mechanism to add the Laplacian random noises to the IoT device training data features, so as to achieve the maximum protection of privacy data. We also present a 5G-enabled edge computing-based framework to train and deploy the model, which can alleviate the computing and communication overhead of the typical IoT systems. Our evaluation results show that APTMFL can efficiently predict subsequent APT behaviors in the IoT system accurately and efficiently.

1. Introduction

With the continuous development of 5G-enabled IoT technologies, numerous mobile applications have emerged with various requirements in terms of intelligence, latency, and bandwidth [1]. However, enormous risks and hidden dangers of information security still exist in the applications of the 5G-enabled Internet of Things (IoT). It is mainly caused by the characteristics of the IoT systems, which are lacking update, having longer lifetimes, delayed patched, and facing consequences of compromise [2, 3]. Among the diverse attacks, Advanced Persistent Threat (APT) belongs to a class of advanced multiple-step attacks. Ascribed to its permeability, concealment, and pertinence, the APT could bring severe threats to the IoT systems [4]. For the sake of defending these increasingly complex and potential security risks, researchers and organizations have put forward

various detection technologies, such as intrusion detection technology, malicious code detection technology, and vulnerability detection technology [5–8]. However, the above-mentioned methods have difficulty in meeting the higher requirements of protection for 5G-enabled IoT systems since APT attacks usually adopt the way of step-by-step penetration and long-term latency to achieve the final purpose of confidential data exfiltration [9].

Recently, the technology of Cyber Situation Awareness (CSA) has been put forward by researchers to solve the above problems. The cyber situation comprehension is a phase of the CSA process that focuses on analyzing the detected malicious activities semantics and the possible internal relationships among them [10]. This kind of technology has the ability to infer the attacker's intention and predict the probability of subsequent attacks, which is quite useful for detecting APT in 5G-enabled IoT systems. Hence, in this

paper, we aim to propose an effective and robust cyber situation comprehension method to predict the probability of subsequent APT attacks occurrence after recognizing APT attacks in the 5G-enabled IoT system.

However, predicting the APT attack in IoT systems could face the following challenges: (1) *Unbalanced Datasets*. Since the APT is a multistep attack model, it is difficult for a single organization to capture the data that can cover the complete APT stages and sufficient attack patterns [11]. In addition, since the different organizations will face different APT attacks, it will contribute to the imbalance of APT data. (2) *Isolated Data Island*. As the data generated by a single organization are not sufficient to describe the complex APT process, integrating the data from several organizations to train a sharing model is a promising way to defend against APT [12, 13]. (3) *Limited Resources of IoT Devices*. The IoT devices are usually resource-constraint where their storage capacity and computing power are usually limited. It is not feasible to assign the large-scale data analysis and process to the IoT devices directly [14]. (4) *Arising Privacy Issues*. Conventional APT prediction methods all need to collect the private information of each device, such as the system logs and device IDs, which could arise significant privacy challenges [15].

To meet the problems above, we proposed an APT Prediction Method based on Differentially Private Federated Learning (APTPMFL) to predict the probability of subsequent APT attacks occurring in IoT scenarios. The contributions we have made are shown as follows:

- (i) We proposed a novel APT prediction method, named APTMFL, which utilizes the federated learning framework to aggregate suspicious activities in the IoT systems. The IoT devices can unite with edge servers to train the prediction model locally using system logs, just uploading the parameter updates to the security service cloud.
- (ii) To protect IoT device data privacy against untrusted edge servers, we adopt a differentially private data perturbation mechanism to perturb the Laplacian random noises to the IoT device training data features, so as to achieve the privacy-preserving property of users' training data.
- (iii) We present an edge computing-based framework to deploy the prediction model in typical IoT systems. The edge servers can not only share the computing overhead for the IoT devices but also alleviate the communication cost between IoT devices and the security service cloud.

The rest of the paper is organized as follows. Section 2 summarizes the related works of attack prediction in cybersecurity and privacy-preserving deep learning. Section 3 provides an overview of the federated learning-based APT prediction architecture for IoT systems, which contains a description of the proposed APTMFL and the edge computing framework for deploying the APT prediction method. Section 4 presents the design details of the APTMFL, which consists of the federated learning

approach and the APT attack prediction. Section 5 shows a view of our experiments and analysis. Section 6 presents some conclusions.

2. Background and Related Work

In 1999, the US Air Force Communications and Information Center originally applied the situation awareness technology to the data fusion analysis of multiple NIDS detection results. They claimed that the multisensor data fusion technology provides an important functional framework for the next generation of the intrusion detection system and CSA [4] system, which can fuse the data of multisource heterogeneous IDS, identify the intruders, attack frequency, threat degree, and so on. The CSA applies the theory and method of situation awareness to the field of cybersecurity so that network security managers can grasp the security status of the dynamic network environment and acquire defending decision support. We also give a general functional model of CSA, as shown in Figure 1. The model includes the cyber situation perception phase, cyber situation comprehension phase, and cyber situation projection phase. The functions of each phase are briefly summarized as follows:

- (i) The function of cyber situation perception is to identify the activities in the system, that is, reduce the noise of the raw data generated by security equipment and information management system to get the valid information and analyze the correlation of them to identify the objects in the system. In this way, the abnormal activities will stand out.
- (ii) Cyber situation comprehension usually focuses on recognizing the malicious activities and correlating the semantics of them. In this way, the attacker's intention can be inferred and the subsequent attacks can also be predicted.
- (iii) Cyber situation projection can analyze and evaluate the threat of attack activities to each object in the information system based on the two steps above. This phase focuses on estimating the effects that attacks have produced and may produce on the objects. By projecting the results of CSA to a certain system object, the state of the object can be obtained in the current situation. Although we want to recognize and analyze the activities, the final result of CSA should be expressed as the influence of these activities on the system objects, not just the identification of activities. Cyber situation projection is a process of feedback understanding, fusing the states of various objects observed from the system to form a situation and then evaluate the significance of the situation to each object.

As an important part of cyber situation comprehension, the attack prediction can analyze the logical relationship between attack behaviors and infer the possible changing trends. The purpose of attack prediction is to infer the subsequent malicious actions by understanding the intention of them. At present, the hot topic of attack prediction is

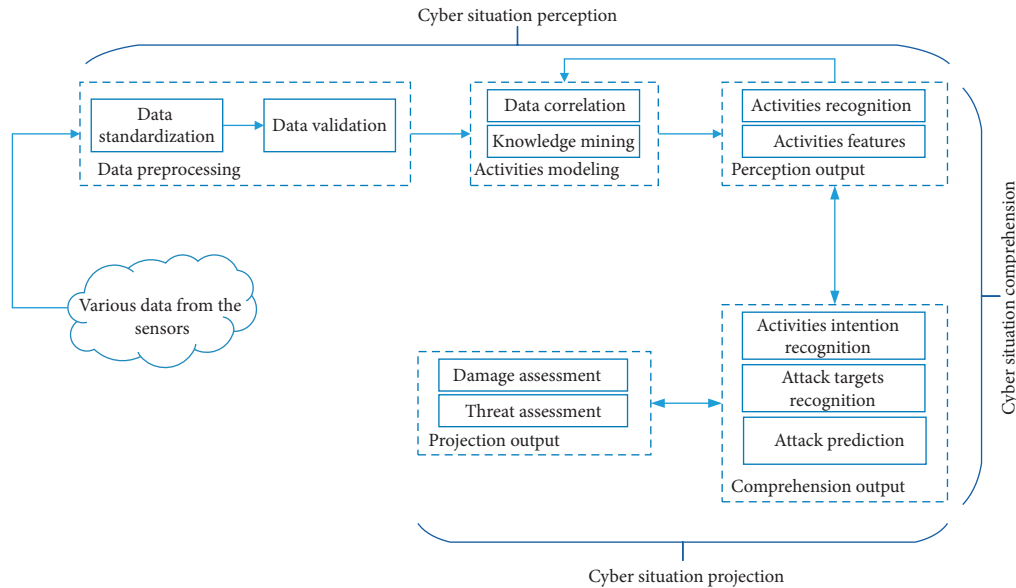


FIGURE 1: The framework of Cyber Situation Awareness.

related to the four topics as follows: attack/intrusion prediction, attack projection, attack intention recognition, and network security situation forecasting [5]. The tasks of attack intention recognition and attack projection are tied to intrusion detection. The core task of them is to predict an adversary's next step moving and his ultimate goal. The attack/intrusion prediction is much more general as it only focuses on predicting malicious activity occurring. The network security situation forecasting is essentially a generic concept related to CSA. The output of network security situation forecasting is a forecast of the number of malicious activities and vulnerability fluctuation in the network.

For solving the challenge of attack activities prediction, Polatidis et al. [6] presented a recommender system that can be applied to defend the cyber threat effectively and practically by making predictions about the ensuing attack behaviors in attack graphs. The Bayesian classifier was developed by Okutan et al. [7], utilized to predict the attack probability in a given day by processing signals extracted from social media and overall events. Huang et al. [8] worked on industrial cyber-physical systems (ICPSs) security and proposed a novel risk assessment approach in virtue of a Bayesian network to model the propagation of malicious activities and predict the probability of IoT devices being attacked. Okutan et al. [9] designed an innovative, automatic attack prediction system called CAPTURE which comprehensively uses generated signals to train a Bayesian classifier used to forecast the cyber threat. Dowling et al. [10] implemented dynamic and adaptive honeypots to capture malicious datasets used to analyze attack types and model temporal attack patterns, and the probability of each attack type proceeding at a certain slot of the day can be calculated. A novel attack prediction method based on information exchange and data mining is presented in [11], which defines rules to describe the general malicious patterns by extracting information from numerous alerts. A system based on

machine learning named MLAPT is suggested in [12]. The proposed system developed eight modules to detect various technics of APT and the machine-learning-based prediction framework takes associated alerts as input to calculate the probability of alerts to evolve a full APT scenario. A data-snapshot-based malware prediction approach is described in [13]. Using recurrent neural networks, this approach can predict malicious executable files in the early stage of software execution. The literature [14] designed a Bayesian game framework based on game theory to analyze multiple APT attack stages and deceptive strategies. Behaviors of APT actors can be predicted by the perfect Bayesian Nash equilibrium (PBNE) to make a defensive strategy. A targeted complex attack network model entitled TCAN is developed in literature [15]. The model predicts the optimum attack path by means of constructing a dynamic attack graph and monitor state change.

Due to the fact that we adopt a differentially private federated learning mechanism to predict APT attacks, we have also surveyed the recent researches on privacy-preserving deep learning. To optimize the efficiency of Deep Neural Networks, the model partition technique [16] has been proposed, which can assign the loosely coupled hidden layers [17] to a third party. Meanwhile, in order to prevent user's privacy data stolen by malicious servers, some researches have been presented from different perspectives [18]. A privacy-preserving deep learning approach proposed by Abadi and Chu [19] provides a Gaussian Perturbation mechanism [20] for the clipped gradient. It is a very effective solution to protect the data privacy of users. To prevent each user's updates from leaking to untrusted third party by learning model sharing, Geyer et al. proposed a device-side differential private federated learning framework [21]. In addition, a secret sharing-based method has been applied to a high-dimensional data sum aggregation protocol by Bonawitz et al. [22]. Unfortunately, all of these privacy

solutions above have to rely on the existence of trusted aggregating servers which work on perturbing the global model parameters and safely assign noise parameters to each user. It means that the aggregator servers can read the model parameters of each individual. Therefore, it is necessary to adopt a practical mechanism to protect the privacy of IoT devices from untrusted third parties in APT attack prediction based on federated learning.

3. APT Prediction for IoT Systems

3.1. System Architecture. In this work, we provide an edge computing framework (shown in Figure 2) that can partition the APTMFL method across the security service cloud, edge servers, and IoT devices. All of the IoT devices take part in the differentially private federated learning protocol provided by a security service cloud, so as to acquire the APT prediction service. Due to malicious edge servers and untrusted security service cloud even have possibility of sneaking into the process of federated learning, to guarantee the involved IoT devices will not suffer the threats of training data breach, we can divide the local learning model into the device-side part and edge-side part which is called ICP-GRU model. The Inception (ICP) models are deployed on IoT devices to extract features from multiple scales of log data. The Gated Recurrent Units (GRU) models are deployed on the edge servers to learn the evolution of APT scenarios. The ultimate goal is not to detect APT attacks but to predict the probability of the evolution of the APT scenario to the next stage which can contribute to the cyber situation comprehension in IoT scenarios. The involved entities and their function are shown in Figure 2.

3.1.1. IoT Devices. The IoT devices represent various kinds of devices (such as industrial personal computers, smart meters, and monitoring equipment). Each IoT device is capable of communication and computing, which can execute the local log data feature extraction procedure (the device-side part ICP model) and transfer its extracted features to the adjoining edge server through 5G base stations.

3.1.2. Edge Server. The edge servers are equipped with much stronger computation and storage resources compared to the IoT devices, usually assigned at the edge of the IoT systems and working as the computation unit between security service cloud and IoT devices. The role of the edge server is a participant in federated learning with the purpose of training the APT prediction model (the edge-side part GRU model). Federated learning is a distributed machine-learning approach with efficient communication and privacy protection. The edge servers receive the features extracted from IoT devices through 5G base stations to train a model locally and update the model parameters to the security service cloud. The participated edge servers can learn various APT attack patterns without exchanging datasets and monitor IoT devices to launch APT attack prediction for cyber situation comprehension.

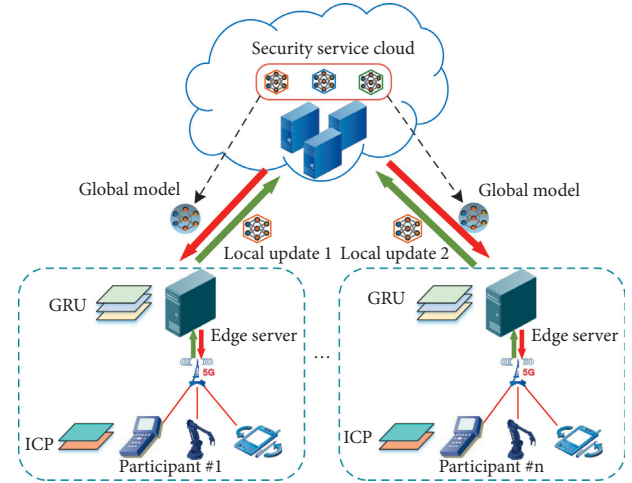


FIGURE 2: System architecture.

3.1.3. Security Service Cloud. Each edge server updates the model parameters to the security service cloud after training the model locally; after that, the security service cloud aggregates the parameters into a global model and assigns the aggregated model to each edge server. Based on that, all participants only need to interact with the security service cloud for model parameters without exchanging their own data, which protects data privacy and improves transmission efficiency. In such a condition, different edge servers could belong to different organizations, and the security service cloud maintains a repository of APT attack patterns for participants. Meanwhile, the challenge of imbalanced APT data can also be alleviated thanks to the distributed learning model.

3.2. Threat Model and Assumptions. Generally, a full APT attack scenario consists of the following attack phases. Initially, an APT performer gains access to the system illegitimately through the point of entry. Then, the attacker will establish a connection with a C&C. After that, the attacker discovers and collects assets within the organization for privilege escalation and lateral movement. Eventually, the adversary will destroy infrastructures or exfiltrate confidential data of the organization to achieve the ultimate goal. During this process, the APT performer will persist for an extended period of time and use numerous technics. The proposed APTMFL focuses on learning the occurring APT scenarios to acquire its evolution features. Under such circumstances, once a new APT alert comes, we can measure the probability of the subsequent malicious activity occurring by inputting the relevant log instances into the ICP-GRU model.

Among these entities involved in the ICP-GRU model, the IoT devices are assumed as the trusted entities which benefit from the security services by collaboratively executing the training process with other IoT devices. Unfortunately, the third parties, that is, security service cloud and edge servers, have the possibility of being honest but curious [23]. In particular, they can faithfully perform the federated

learning process and correctly calculate and send results. However, they are curious about the privacy contained in the log data and try to acquire the privacy data [24]. In addition, we assume that the model will not be poisoned by attackers during the model training process. At last, we assume the integrity of the data collection framework which completely records the operations of the system, and the data will not be falsified by attackers.

4. APTMFL Design

4.1. Federated Learning Approach. In order to realize APT attack prediction in IoT scenario to achieve security situation comprehension, we introduce an efficiently differentially private federated learning model in edge computing IoT system, named ICP-GRU (the learning procedure is shown in Figure 3). Considering performance as well as privacy issues, this machine-learning model is divided into the device-side part and the edge-side part. Thereby, the computation overhead on the IoT devices is also able to be reduced [25]. The ICP-GRU relies on the edge computing architecture in the IoT scenario and splits the learning protocol across the participants. Specifically, the local federated learning training procedure is divided into two phases: device-side ICP and edge-side GRU. According to the division mechanism, the inception convolution layers are assigned to the device-side while the remaining layers (i.e., Gated Recurrent Units layers) are deployed on the edge side. In this approach, IoT devices only extract and perturb the lightweight and simple features of log instances. To guarantee the system log data in rigorous privacy protection, we provide a differential privacy mechanism in our ICP-GRU scheme, where the extracted features from log instances are perturbed by the deliberate Laplace noise before being transmitted to the edge server through 5G base stations. The security service cloud in the ICP-GRU model is designed to aggregate and average the local updates provided by the edge servers.

To provide proper data for this APT prediction approach based on differentially private federated learning, we also standardize and normalize the original system log. Since the attributes of some features are character types, such as pname, q_domain, and referer, all the symbolic features needed to convert into numerical types before feeding the dataset into the neural network. At the same time, the value of each feature dimension is inconsistent, and the range of values is also obviously different. Some data with high values on high-magnitude features perhaps have a large weight, thus ignoring some hidden information on low-level data. Therefore, we provide a log instance construction module at the beginning of the ICP-GRU model to preprocess the raw data.

4.1.1. Inception Convolution and Data Perturbation. As CNN achieved excellent performance in image processing, it is also constantly being applied to other fields. However, traditional CNN only focuses on extracting local features and neglects the aggregation of multiple local features. To flexibly mitigate this problem, Google proposed a

convolutional neural network architecture in the GoogleLeNet network called Inception. The Inception module aggregates $1 * 1$, $3 * 3$, and $5 * 5$ convolution kernels and max-pooling into one layer. Multiple convolution kernels extract information of different scales of the dataset, and the fusion of features can obtain a better representation of the log instance. We adopt the Inception module to perform convolution operation which extracts features from multiple scales of the log instances dataset which can make the neural network much smarter.

After data preprocessing, the dataset arranged according to timestamp is fed into the inception convolution module in the form of data flow for training. Considering the contextual correlation between data, the data flow is split into fixed-size vectors, and each vector contains n pieces of data. After processing each vector, an $n * m$ feature matrix can be formed where m represents the number of features of each data. The inception convolution module will extract the features of the dataset through convolution kernels of $1 * 1$, $2 * 2$, and $3 * 3$ and max-pooling of $2 * 2$, and the same convolution should be utilized in order to match the width and height of the output matrix. Nevertheless, the Inception module is resource-consuming when performing the convolution operation. Therefore, the $1 * 1$ convolution is inserted before $2 * 2$ and $3 * 3$ convolution to reduce the feature dimension and increase the computation speed.

As the federated learning approach usually trains the sensitive data locally, it can provide a basic privacy guarantee to the involved IoT devices. However, these sensitive device data such as updated parameters (i.e., features and gradients) still have probability stolen by the untrusted security service cloud and edge servers. Therefore, it is urgent to formulate a robust preserving mechanism to keep the confidentiality of each IoT device against the untrusted security service cloud and edge servers. Thereby, we perturb the features extracted by the ICP model, so as to protect the privacy of the IoT device logs. To meet the aforementioned challenge, we formulate a differentially private data perturbation mechanism to defense the untrusted entities acquiring the privacy information that exists in the features extracted by the IoT device-side ICP model. The ICP model can be regarded as a deterministic function: $x_l = F(x_r)$, where x_r is the private device log data and x_l represents the l -th layer output of the ICP model. In order to achieve privacy protection, the differential privacy approach is applied to the ICP model and our private federated learning protocol is constructed in the edge computing-based IoT scenario. As an efficient means to achieve ϵ -differential privacy, the controlled Laplace noise is added to the extracted features. The Laplace noise is sampled from the Laplace distribution with scale $(\Delta F/\epsilon)$ into the output x_l . According to the definition of differential privacy, the global sensitivity for a query $f: D \rightarrow R$ can be defined as follows:

$$\Delta f = \max_{d \in D, d' \in D'} |f(d) - f(d')|. \quad (1)$$

4.1.2. GRU Networks. The GRU is a type of RNN that is similar to LSTM, which is proposed to address the problems of long-term memory and gradient in backpropagation. The

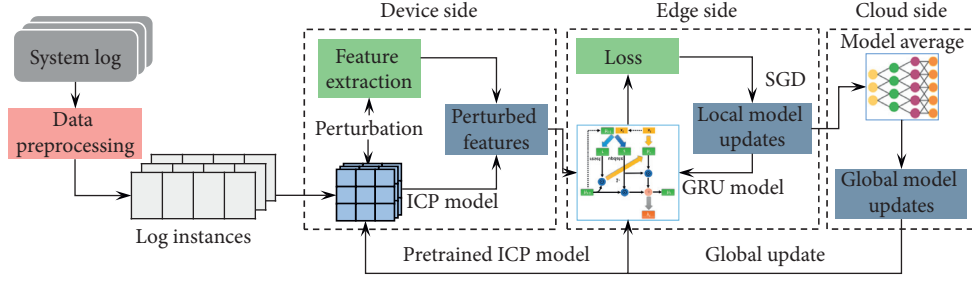


FIGURE 3: ICP-GRU learning process.

reason for adopting GRU is that it not only achieves the effect equivalent to LSTM but also can save more computing resources, which can greatly improve training performance.

Similar to RNN, the hidden state h^{t-1} transmitted from the previous node and the current input x^t constitute the inputs of GRU. Combining x^t and h^{t-1} , GRU will get the output of the current node as y^t and the hidden state h^t transmitted to the next node. Initially, two gates are obtained by the previous state h^{t-1} and the current node input x^t . As shown in formulas (2) and (3), r indicates the reset gate, z represents the update gate, and σ represents the *Sigmoid* function.

$$r = \sigma(W^r x^t + U^r h^{t-1}), \quad (2)$$

$$z = \sigma(W^z x^t + U^z h^{t-1}). \quad (3)$$

After getting the gate signal, the reset gate is used to get the reset data $h^{t-1} \odot r$, that is, h^{t-1} . Then input h^{t-1} and x^t into the $\tanh(\cdot)$ activation function so as to the output range will fall in $[-1, 1]$. h' here mainly involves the currently entered data x^t , and adding h' to the current hidden state in a targeted manner is equivalent to memorizing the state at the current moment. The formal description of h' is shown as follows:

$$h' = \tanh(W^h x^t + U^h (h^{t-1} \odot r)). \quad (4)$$

Finally, the GRU carries out two operations (forgetting and memorizing) at the same time. With the acquired update gate whose value range falls in $[0, 1]$, $(1 - z) \odot h^t$ defines how much the previous memory is forgotten, and $z \odot h'$ defines how much of the h' containing the current node information is to be keep around. The more close the update gate to "1" is, the more the memory will be reserved, and the more close the update gate to "0" is, the more the memory will be forgotten. The formal description of h_t is shown as follows:

$$h_t = (1 - z) \odot h^t + z \odot h'. \quad (5)$$

Multiscale features hidden in the data flow can be extracted after the ICP model is processing plenty of log instances. The features are subsequently fed into the GRU network in serialized form to learn the temporal features by selectively learning and forgetting. The model parameters will be continuously updated by means of gradient back-propagation, and a powerful APT attack prediction model will be obtained in the wake of multiple rounds of iterations.

4.1.3. Federated Learning Procedure. The APTMFL method proposed in the paper is applied to the IoT edge computing environment. As shown in Figure 4, the federated learning process for APT prediction consists of the following steps:

- (1) The edge servers act as participants in federated learning and request the ICP-GRU model from the security service cloud.
- (2) The security service cloud assigns an initialized model to each edge server once the participants' requests are received, and the corresponding ICP model is transmitted to the connected IoT devices through 5G communication.
- (3) Each IoT device inputs locally collected data into the model for training, learning the private log data independently, and perturbs the features by controlled Laplace noise.
- (4) The edge servers get the features transmitted from the IoT devices for further GRU learning and update the local model parameters to the security service cloud once achieving model training.
- (5) The security service cloud aggregates the local updated models into a global model.
- (6) The security service cloud will deliver the aggregated global model to each edge server again. After multiple rounds of retraining, the global model is aggregated by the security service cloud, and the APT attack patterns will be acquired.

Eventually, the security service cloud delivers the global model to each participant which is employed to predict the process of APT attackers in the 5G-enabled IoT system. The pseudocode of the overall APTMFL method is presented in Table 1 and the notations used in the pseudocode are listed in Table 2.

4.2. APT Attack Prediction. APT attackers penetrate the target 5G-enabled IoT system for an extended period of time and launch attacks slowly making defenders have to monitor the system behavior incessantly, which brings great challenges to process data efficiently and detect attacks accurately. However, this low-and-slow attack is a double-edged sword. The long span of time between the APT attack phases leaves enough time for defenders to predict the attacker's next move. When an APT attack at a certain stage is detected

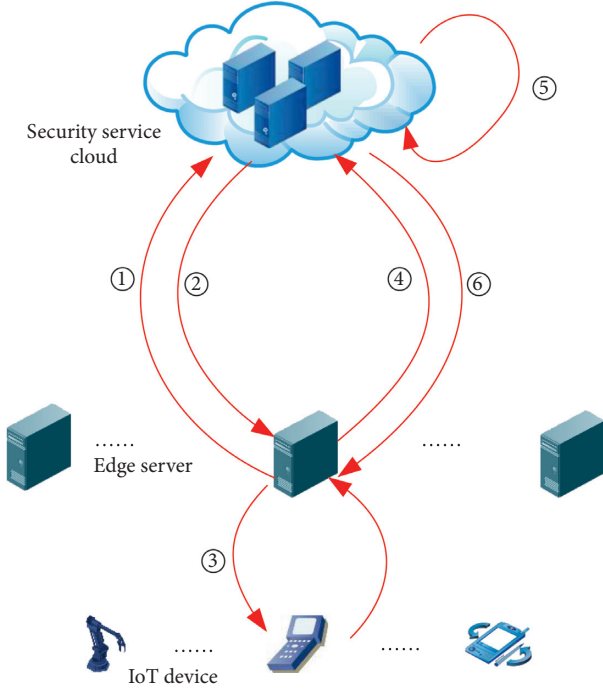


FIGURE 4: Federated learning process.

TABLE 1: The APTMFL pseudo-code.

APTMFL algorithm	
(1)	procedure IoT DEVICE SIDE
(2)	Nullification operation: $x_r' = x_r \odot I_n$
(3)	Features extraction: $x_i = F(x_r')$
(4)	Norm bounding: $x_i' = x_i / \max(1, x_{i\infty} / B)$
(5)	Adding perturbation: $\bar{x}_i = x_i' + \text{Lap}(B/\delta I)$
(6)	Upload $\bar{x}_i \leftarrow \bar{x}_i$ to the edge server
(7)	end procedure
(8)	procedure EDGE SERVER SIDE
(9)	EVENT: Receive perturbed features \bar{x}_i
(10)	Initialize: $w_t^{(i)} \leftarrow w_t^{(\text{global})}$
(11)	for each $E_k \in (1, \dots, E)$ do
(12)	for batch $b \in B$ do
(13)	$g_t^{(i)} = \nabla_{w_t} L(w_t, \bar{x}_i)$
(14)	$w_{t+1}^{(i)} = w_t^{(i)} - \sigma \cdot g_t^{(i)}$
(15)	end for
(16)	$w_{t+1}^{(i)} = w_{t+1}^{(i)} - w_t^{(i)}$
(17)	end for
(18)	Upload $\Delta w_{t+1}^{(i)}$ to the SECURITY SERVICE CLOUD
(19)	end procedure
(20)	procedure SECURITY SERVICE CLOUD SIDE
(21)	EVENT: Receive local model updates $\Delta w_{t+1}^{(i)}$
(22)	Initialize global parameters $w_t^{(\text{global})}$
(23)	for all $i \in [1, n]$ do
(24)	$w_{t+1}^{(\text{global})} = w_t^{(\text{global})} + (1/n) \sum_{i=1}^n \Delta w_{t+1}^{(i)}$
(25)	end for
(26)	Send $w_{t+1}^{(\text{global})}$ to the edge server
(27)	end procedure

and the corresponding alerts are generated, suspicious logs can be continuously analyzed to predict the attacker's behavior so that the proper defensive measures can be chosen before the APT attackers achieve their ultimate goal. In this

TABLE 2: The notations used in the pseudocode.

Notation	Definition
x_r	IoT device sensitive raw data
\bar{x}_r	Nullification result for x_r
\bar{x}_i	Perturbed features
I_n	Nullification matrix
x_i	1-th layer output of DNN
x_i'	Nullification result for x_i
l	Partitioned layer
$F(x_r)$	Deep neural network
$L(w_t, \bar{x}_i)$	Objective loss function
B	Norm bounding
E	Local epoch
δ	Local learning rate
$w_t^{(i)}$	Parameters for IoT device i at t
$g_t^{(i)}$	Gradient for IoT device i at t
$\Delta w_t^{(i)}$	Local model update at t
$w_t^{(\text{global})}$	Global model parameters at t
σ	Noisy scale

section, we first provide the log instance construction procedure for APT behavior learning and prediction. Then, we present an APT prediction procedure.

4.2.1. Log Instance Construction. The log data collected on the IoT devices infected by APT attacks are used for model training and APT attack prediction. The detection system will generate a series of alerts when a certain APT attack step is detected in the 5G-enabled IoT systems. Although the recognized APT attacks have triggered the alerts, there are still plentiful related unaggressive malicious behaviors that have not been detected which could be a stepping stone for the attacker to launch the next step attack activity.

Consequently, alert attribute values will be analyzed to discover the threatened IoT devices after some alerts have emerged. The log data generated in the targeted IoT devices will be constructed into log instances. The types of log instances depend on the data providers and the operating system installed in IoT devices. On the assumption that all log data derive from Windows Embedded Compact (Windows CE) IoT devices, various types of logs provided by different application programs or record facilities are shown in Table 3. The log data are supposed to transform into a uniform format for the purpose of processing data effectively. As shown in Table 4, 14 features are selected from the log data to construct the log instance. Due to the fact that different log data sources and not all features are contained in each log instance, each log instance can be described as a 14-tuple: $A(I(\log)_m) = (a_1, a_2, \dots, a_{13}, a_{14})$. If a log instance only has features a1, a3, and a5, the other feature values are set to zero.

4.2.2. APT Prediction Procedure. Once the ICP-GRU model has accomplished learning the APT attack activity behaviors in the target 5G-enabled IoT system, it can be used for predicting the probability of later APT attack activities. We abstract 4 APT stages under the IoT environment, that is,

TABLE 3: Log types from different recorders.

Type	Logs	Recorders
1	Object access logs	Audit
2	Process create logs	Audit
3	WFP connect logs	Audit
4	HTTP logs	Internet explorer
5	DNS logs	Tshark
6	Authentication logs	Syslogd

TABLE 4: Log instance features extracted from log records.

Type	Logs	Features	Annotation
1	Log3	h_ip	Host IP address
2	Log3	d_ip	Destination IP address
3	Log3	h_port	Host port number
4	Log3	d_port	Destition port number
5	Log3	Type	Request/response
6	Log5	q_domain	D queried domain name
7	Log5	r_ip	DNS resolved IP address
8	Log2	Ppid	Base-16 parent process ID
9	Log1 Log2 Log3	Pid	Base-16 process ID
10	Log1 Log2 Log3 Log6	Pname	Process
11	Log1	Objname	Object name
12	Log4	res_code	Response code
13	Log4	Referer	Refer of requested URI
14	Log1-Log6	Timestamp	Event timestamp

point of entry, C&C communication, asset/data discovery, and data exfiltration. The log instances can be classified into benign and suspicious through the Log Instance Community Detection algorithm proposed in our previous work [13], and only suspicious instances can be collected to train the model. The overall APT prediction process is presented in Figure 5.

Suppose that the APT stages i and $i + 1$ are detected at the time of T_1 and T_2 , respectively; all of the suspicious log instances within the time window $[T_1 - \tau, T_2]$ should be fed to the ICP-GRU model to learn attack behavior features. It means that when the IoT devices suffer from these suspicious activities, the APT attacker will conduct the $i + 1$ stage attack with a high probability. In case a certain APT stage is detected at time T_1 and corresponding alerts are triggered, log instances of threatened devices are recorded starting at $T_1 - \tau$. The suspicious log instances are fed into the prediction model in real time, and the output of the model is a probability value in the range of $[0, 1]$ which indicates the probability of the system suffering the next stage APT attack. Once the output value surpasses the prediction threshold λ , the 5G-enabled IoT system will be in danger of the next stage of an APT attack scenario with high probability.

5. Experimental Evaluation

For the sake of carrying on a detailed and objective evaluation of our proposed APT prediction method (APTPMFL), we implement a federated learning prototype

on the system log data which are generated within the typical seven APT attack scenarios (Op-Clandestine Fox, Hacking Team, APT on Taiwan, Tibetan and HK, Op-Tropic Trooper, Russian Campaign, and Attack on Aerospace) [26].

5.1. Datasets and Experimental Setup

5.1.1. Datasets. It is unfortunate that the appropriate system log dataset and attack alert dataset associated with typical APT attacks are not acquirable. However, our previous work [26] has accomplished the construction of the APT scenario and log instance correlation. Therefore, we adopt the labeled log instances and recognized APT scenarios generated in this work as simulated data to evaluate the effectiveness of APTPMFL. Table 5 presents the details about these datasets. Each dataset is generated by reconstructing a kind of typical APT scenario. These APT scenarios are designed and launched by different APT attack teams; the attackers exploit various vulnerabilities and adopt different attack strategies. We provide these various datasets that can adequately verify the predicting effect of our method for different APT scenarios.

5.1.2. Experimental Settings. In order to make the laboratory environment reflect the characteristics of the real IoT system as similar as possible, we link four identical hosts (a Red Hat Linux operating system running on a host with an Intel Core i7-8550u 2.53 GHz CPU, 16 GB RAM) to deploy an edge computing network based on federated learning. The laboratory environment is shown in Figure 6. The first host works as the security service cloud, and the other three hosts work as the edge servers. The reason for deploying the identical computing resource on the three hosts is that we, respectively, set eight virtual machines in the victim edge servers as virtual IoT devices. Half of the edge servers' computing resource is shared by the virtual IoT devices. It means that an edge server owns 8 GB RAM, and each virtual IoT device occupies 1 GB RAM. This resource allocation scheme is very consistent with the real IoT system. The APTPMFL is proposed to meet the challenge of the IoT device's resource limitation. Even though we do not implement the real IoT operation flow, each virtual IoT device just has 1 GB RAM resource and without any updated patches can competently simulate and testify the efficiency of our method in the IoT system.

To accomplish the federate learning training process of APTPMFL, we allocate the labeled log instances to each virtual machine (simulates the IoT devices) and allocate the recognized APT scenarios to the edge servers. Then, the virtual machines will transmit part of the logs to the edge servers for training the ICP-GRU models and keep the other logs for testifying the prediction performance. As the baseline of the experiments, we adopt the standard federated learning approach presented in the literature [27]. The local training configurations are that the prediction train for $E = 20$ local epochs with the initial learning rate $\delta = 0.1$. Each module of APTPMFL works on the corresponding locations based on the proposed edge computing-based framework.

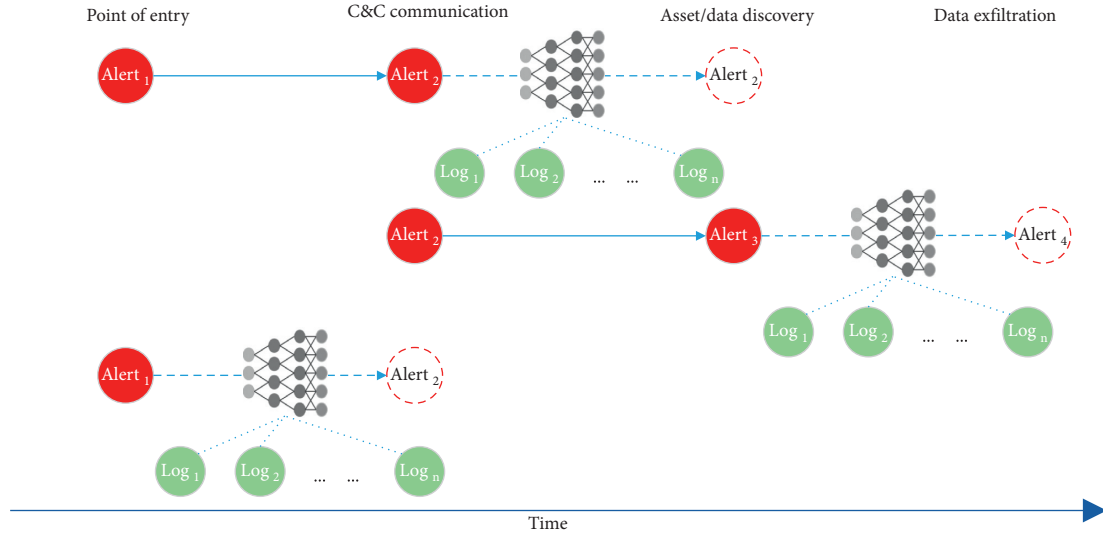


FIGURE 5: APT prediction process.

TABLE 5: Datasets used in our experiments.

Dataset	Input size	Training samples	Testing samples
Op-Clandestine Fox	14 × 1	5600	1200
Hacking Team	14 × 1	5800	1240
APT on Taiwan	14 × 1	5750	1232
Tibetan and HK	14 × 1	6300	1350
Op-Tropic Trooper	14 × 1	6100	1307
Russian Campaign	14 × 1	7800	1671
Attack on Aerospace	14 × 1	4900	1050

Finally, we verify the performance of our method based on some evaluation indicators.

The essence of proposing the APTMFL is to predict the probability of subsequent APT attacks occurring in IoT scenarios. We implemented the algorithm and federated learning framework on the laboratory edge servers and security service cloud. For the sake of verifying whether the proposed method can effectively predict the probability of subsequent APT attacks occurring in the laboratory environment, we select the $F1$ score and false-positive rate (FPR) to indicate the performance of APTMFL. The reason for adopting the $F1$ score instead of the common indicators $Recall = TP / (FN + TP)$ and $Precision = TP / (TP + FP)$ is that the two indicators above are mutually exclusive in some cases, needing a harmonic mean to balance respective defects. The parameters TP , FN , and FP , respectively, count the number of true-positive prediction probabilities, the number of false-negative prediction probabilities, and the number of false-positive prediction probabilities. Thereby, the formal description of the F_1 -score is shown in formula (5). The FPR focuses on representing the proportion of false alert of the organization that is in danger of the next APT attack stage. The formal description of FPR is shown in formula (6).

$$F_1 = \frac{2Recall \cdot Precision}{Recall + Precision}, \quad (6)$$

$$FPR = \frac{FP}{FP + TN}. \quad (7)$$

5.2. Evaluation of APTMFL

5.2.1. APT Prediction Performance. We will evaluate the prediction performance of the APTMFL by analyzing the results of the FPR and F_1 -scores. The prediction threshold λ can influence the prediction result as more next step APT attack alerts will generate with its value higher. We have evaluated the prediction performance of APTMFL on the 7 typical APT attack scenarios, such as Op-Clandestine Fox, Hacking Team, APT on Taiwan, Tibetan and HK, Op-Tropic Trooper, Russian Campaign, and Attack on Aerospace. The corresponding system logs are reconstructed by one of our previous works [26].

The performances of APTMFL predicting the 7 typical APT attacks are shown in Figures 7 and 8. It is easy to get the result that both the FPR and F_1 will reduce with the value of prediction threshold λ increasing. It is due to the fact that the lower λ will make more log instances detected as prestep of APT attack activities and the benign log instances will have a higher possibility to be incorrectly detected. We work hard for getting a proper threshold to make our method achieve preferable prediction performance on the 7 typical APT attack scenarios. Fortunately, when the value of the prediction threshold λ is 0.75, the F_1 -scores are not too low (at around 80%) and the FPR can drop to an acceptable level (not exceeding 5%).

5.2.2. Efficiency of Federated Learning. We have conducted a set of experiments to evaluate the federated learning performance with varying the quantity of IoT devices (from 3 to 24). The number of epochs that each IoT device collaborates

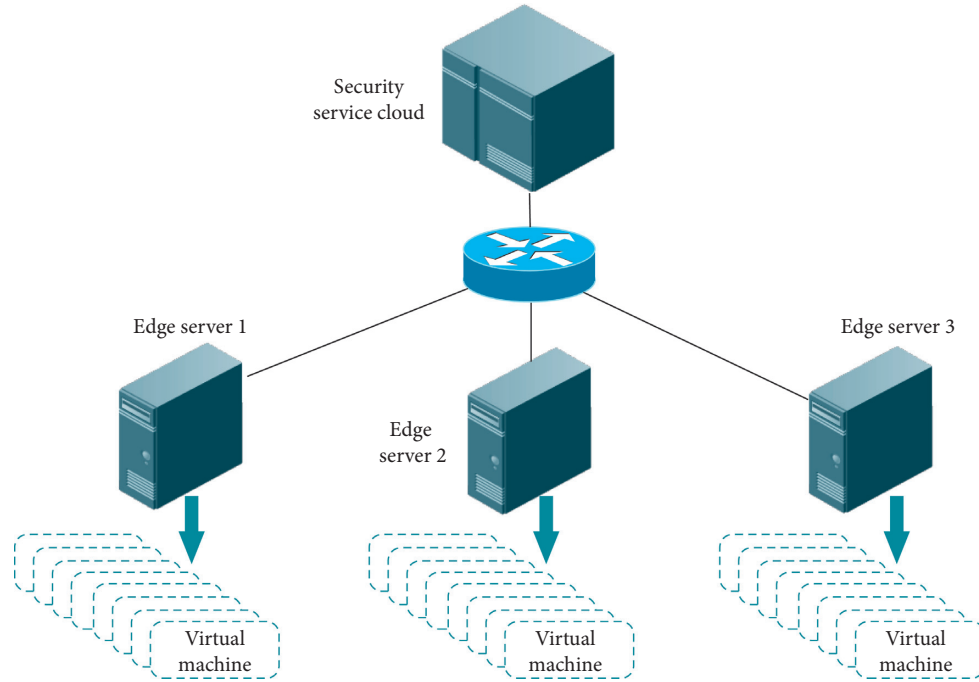
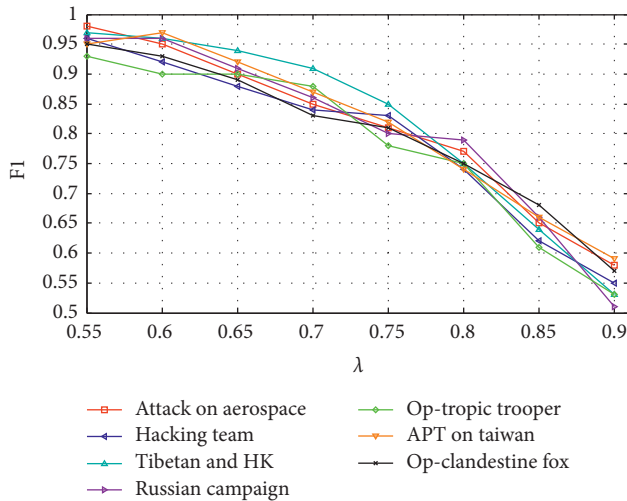
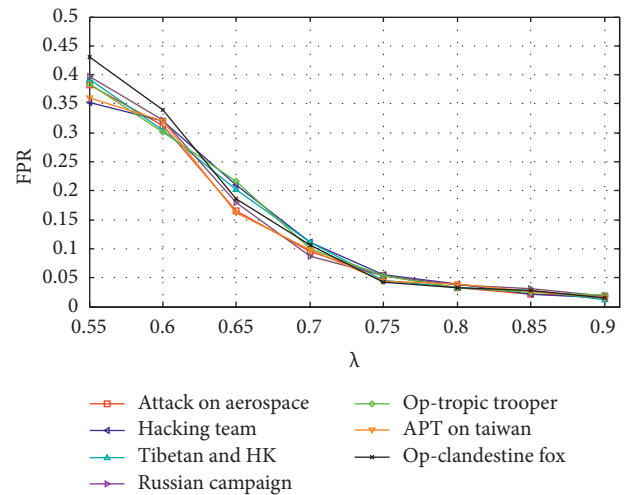


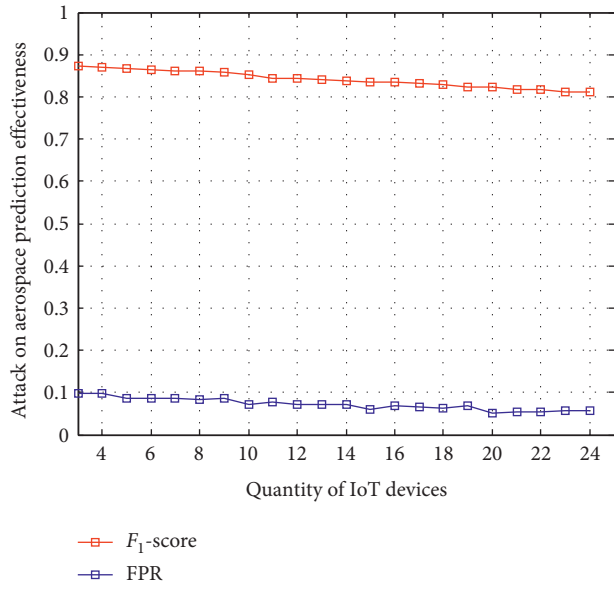
FIGURE 6: Laboratory environment setup.

FIGURE 7: The F_1 varies with threshold λ fluctuation.FIGURE 8: The FPR varies with threshold λ fluctuation.

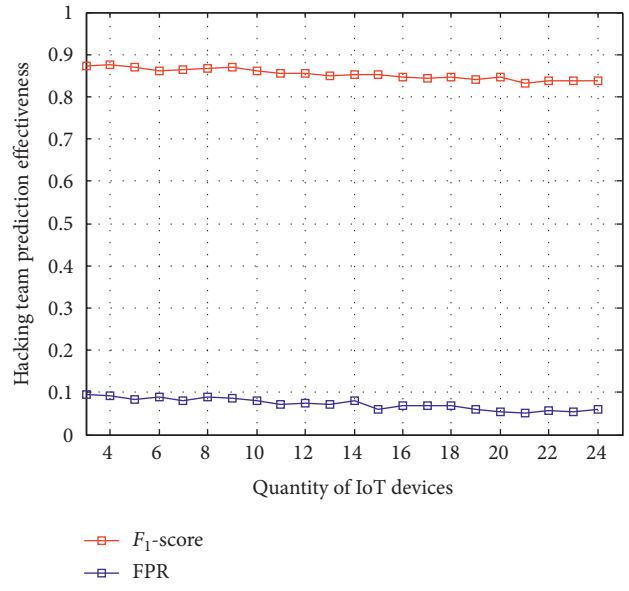
with edge server training the ICP-GRU local model is set as 20 and the number of communication rounds between edge server and security service cloud is set as 5. Thereby, each local model has been trained in a total of 100 epochs. It is a sufficient number of training epochs due to the fact that we have accomplished training the ICP-GRU model in a centralized learning scenario and got convergence after 98 epochs. We allocated a randomized subset of training IoT device logs from the constructed dataset to each virtual IoT device and the proposed APTMFL method performance has been evaluated with the number of devices involved in the federated learning model varying. We repeated this experiment seven times for each APT scenario, with random resampling of the training

datasets. The value of prediction threshold λ is set as 0.75 to balance the F_1 -score and FPR. The evaluation results of seven APT scenarios are shown in Figure 9 which demonstrates that the APTMFL method with more participating IoT devices can acquire better FPR and the F_1 -score deteriorates only slightly. Besides, we also find another feature of APTMFL which is that the time complexity will approximately linearly increase with the quantity of log instance increasing.

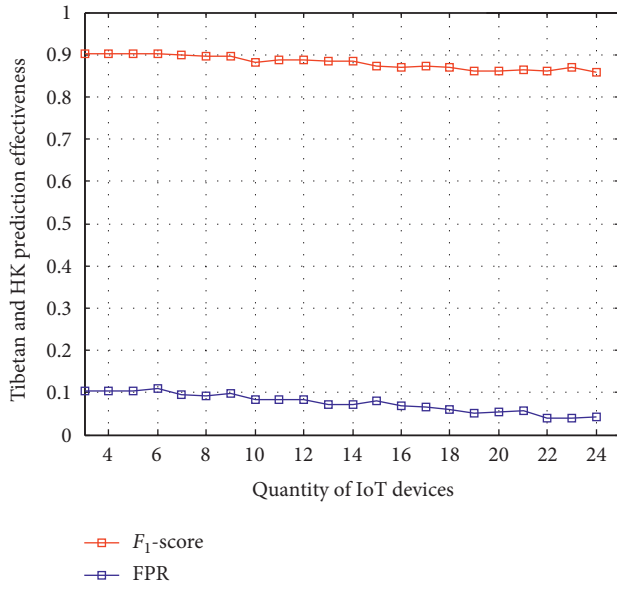
The proposed APTMFL method can provide better privacy for IoT devices contributing to without training data sharing during the training procedure. However, the ICP-GRU model still has some limitations. Comparing with training this model in a centralized framework, it will



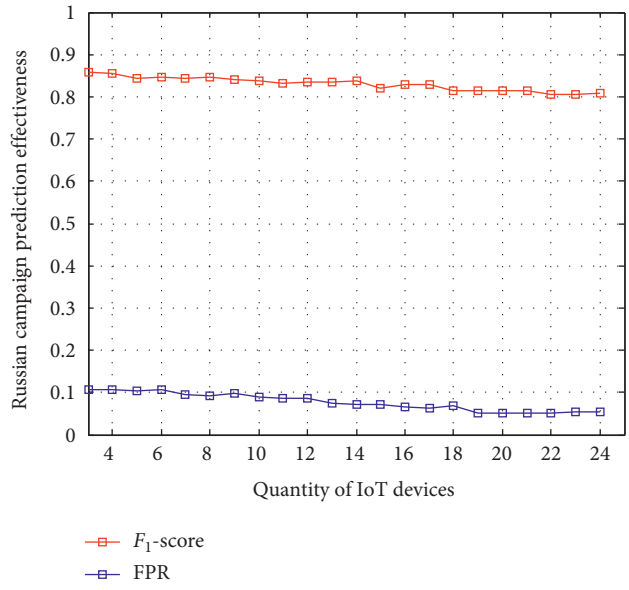
(a)



(b)



(c)



(d)

FIGURE 9: Continued.

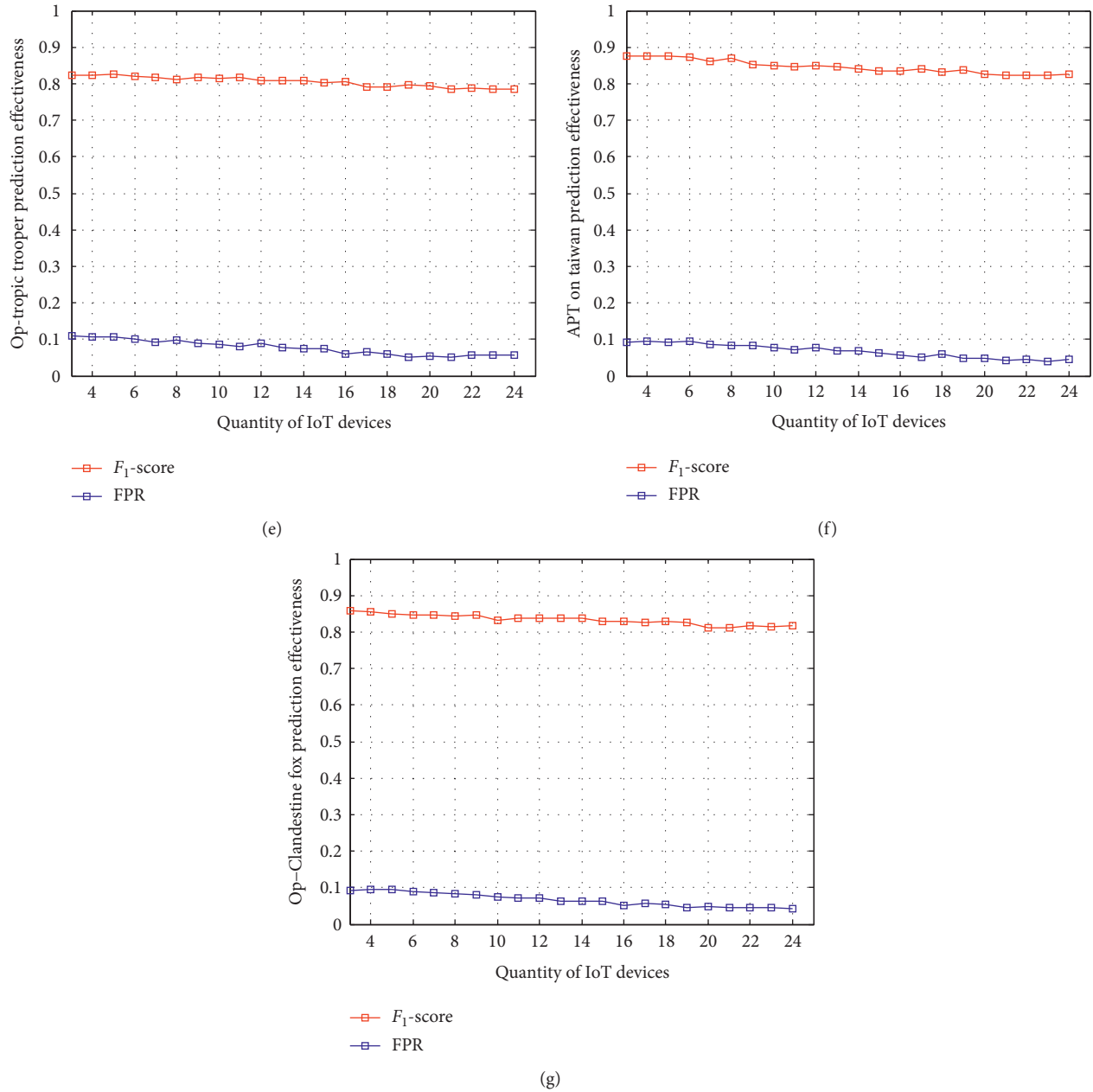


FIGURE 9: Evolution of F_1 and FPR as we increase the number of IoT devices. (a) Attack on Aerospace. (b) Hacking Team. (c) Tibetan and HK. (d) Russian Campaign. (e) Op-Tropic Trooper. (f) APT on Taiwan. (g) Op-Clandestine Fox.

TABLE 6: Effect of using federated learning compared to centralizing approach.

Metric	Centralized learning (%)	Federated learning			
		6 participants (%)	12 participants (%)	18 participants (%)	24 participants (%)
F_1 -score	89.8	88.7	86.5	84.5	81.4
FPR	4.85	4.87	4.89	4.94	4.96

inevitably lose some accuracy of APT prediction. To compare this inevitable loss in accuracy, we set another experiment to retrain four ICP-GRU models using the entire training dataset by dividing it among 6, 12, 18, or 24 IoT devices and comparing these to an ICP-GRU trained in a

centralized framework. The evaluation values of F_1 -scores and FPR are the average of their effectiveness on 7 APT scenarios when the threshold λ is set as 0.75. Table 6 shows a small decrease in F_1 -scores as we increase the number of IoT devices, and the FPR does not fluctuate evidently. This small

drop in F_1 -scores will not be concerned, because we can fix the threshold λ to amend it to an appropriate level.

6. Conclusions

We present APTPMFL, a federated learning-based APT prediction method deployed on the 5G-enabled IoT scenario. A model containing multiple APT attack patterns is trained in a distributed learning manner and the well-trained model will be implemented to predict the probability of subsequent APT attacks occurring in 5G-enabled IoT scenarios. As a result, the experiments show that APTPMFL successfully predicts the probability of subsequent APT activities with acceptable accuracy and low false rates.

Data Availability

As this work was supported by the National Key Research and Development Program of China which involves security and secrecy in the military domain, the datasets are not suitable for public exposure.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the National Key Research and Development Program of China, under Grant no. 2019YFB2102000.

References

- [1] K. Palani, E. Holt, and S. Smith, "Invisible and forgotten: zero-day blooms in the IoT," in *Proceedings of the IEEE International Conference on Pervasive Computing & Communication Workshops*, pp. 1–6, Melbourne, Australia, March 2016.
- [2] M. Marchetti, F. Pierazzi, M. Colajanni, and A. Guido, "Analysis of high volumes of network traffic for advanced persistent threat detection," *Computer Networks*, vol. 109, pp. 127–141, 2016.
- [3] P. Hu, H. Li, H. Fu, D. Cansever, and P. Mohapatra, "Dynamic defense strategy against advanced persistent threat with insiders," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, pp. 747–755, Hong Kong, China, March 2015.
- [4] T. Bass, "Intrusion detection systems and multisensor data fusion," *Communications of the ACM*, vol. 43, no. 4, pp. 99–105, 2000.
- [5] M. Husák, J. Komárková, E. Bou-Harb, and P. Celeda, "Survey of attack projection, prediction, and forecasting in cyber security," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 640–660, 2019.
- [6] N. Polatidis, E. Pimenidis, M. Pavlidis, and H. Mouratidis, "Recommender systems meeting security: from product recommendation to cyber-attack prediction," *Engineering Applications of Neural Networks*, vol. 11, no. 3, pp. 508–519, 2017.
- [7] A. Okutan, S. J. Yang, and K. McConky, "Predicting cyber attacks with Bayesian networks using unconventional signals," in *Proceedings of the 12th Annual Conference on Cyber and Information Security Research*, pp. 1–13, Oak Ridge, TN, USA, April 2017.
- [8] K. Huang, C. Zhou, Y.-C. Tian, S. Yang, and Y. Qin, "Assessing the physical impact of cyberattacks on industrial cyber-physical systems," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 10, pp. 8153–8162, 2018.
- [9] A. Okutan, G. Werner, K. McConky, and S. J. Yang, "POSTER: cyber attack prediction of threats from unconventional resources (CAPTURE)," in *Proceedings of the ACM SIGSAC Conference*, pp. 2563–2565, Dallas, TX, USA, October 2017.
- [10] S. Dowling, M. Schukat, and H. Melvin, "Using analysis of temporal variances within a honeypot dataset to better predict attack type probability," in *Proceedings of the International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 349–354, London, UK, December 2017.
- [11] M. Husák and J. Kašpar, "Towards predicting cyber attacks using information exchange and data mining," in *Proceedings of the International Wireless Communications Mobile Computing Conference (IWCMC)*, pp. 536–541, Caen, France, July 2018.
- [12] I. Ghafir, M. Hammoudeh, V. Prenosil et al., "Detection of advanced persistent threat using machine-learning correlation analysis," *Future Generation Computer Systems*, vol. 89, no. 1, pp. 349–359, 2018.
- [13] M. Rhode, P. Burnap, and K. Jones, "Early-stage malware prediction using recurrent neural networks," *Computers & Security*, vol. 77, no. 2, pp. 578–594, 2018.
- [14] L. Huang and Q. Zhu, "Adaptive strategic cyber defense for advanced persistent threats in critical infrastructure networks," *Performance Evaluation Review*, vol. 46, no. 2, pp. 52–56, 2018.
- [15] W. Niu, X. Zhang, G. Yang, R. Chen, and D. Wang, "Modeling attack process of advanced persistent threat using network evolution," *IEICE Transactions on Information and Systems*, vol. E100-D, no. 10, pp. 2275–2286, 2017.
- [16] S. A. Osia, A. S. Shamsabadi, and A. Taheri, "A hybrid deep learning architecture for privacy-preserving mobile analytics," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, pp. 1–21, 2017.
- [17] J. Dean, G. Corrado, R. Monga, and K. Chen, "Large scale distributed deep networks," in *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1223–1231, Lake Tahoe, NV, USA, December 2013.
- [18] J. L. Zhang, J. Wang, Y. C. Zhao, and B. Chen, "An efficient federated learning scheme with differential privacy in mobile edge computing," in *Proceedings of the 4th EAI International Conference on Machine Learning and Intelligent Communications (MLICOM)*, pp. 538–550, Nanjing, China, August 2019.
- [19] M. Abadi and A. Chu, "Deep learning with differential privacy," in *Proceedings of the 23th ACM Conference on Computer and Communications Security (CCS)*, pp. 308–318, Vienna, Austria, October 2016.
- [20] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2013.
- [21] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: a client level perspective," in *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1–7, Montréal, Canada, December 2018.
- [22] K. Bonawitz, V. Ivanov, B. Kreuter, and A. Marcedone, "Practical secure aggregation for privacy-preserving machine

- learning,” in *Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS)*, pp. 1175–1191, Dallas, TX, USA, October 2017.
- [23] H. Li, K. Ota, and M. Dong, “Learning IoT in edge: deep learning for the internet of things with edge computing,” *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.
- [24] S. Chang and C. Li, “Privacy in neural network learning: threats and countermeasures,” *IEEE Network*, vol. 32, no. 4, pp. 61–67, 2018.
- [25] J. L. Zhang, Y. C. Zhao, and J. Wang, “FedMEC: improving efficiency of differentially private federated learning via mobile edge computing,” *Mobile Networks and Applications*, vol. 3, pp. 1–13, 2020.
- [26] X. Cheng, J. Zhang, and B. Chen, “Cyber situation comprehension for IoT systems based on APT alerts and logs correlation,” *Sensors*, vol. 19, no. 18, p. 4045, 2019.
- [27] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proceedings of the AISTATS*, pp. 1–10, Lauderdale, FL, USA, April 2017.

Research Article

SANS: Self-Sovereign Authentication for Network Slices

Xavier Salleras ^{1,2} and Vanesa Daza ^{1,2}

¹Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain

²Center for Cybersecurity Research of Catalonia (CYBERCAT), Catalonia, Spain

Correspondence should be addressed to Xavier Salleras; xavier.salleras@upf.edu

Received 31 July 2020; Revised 1 October 2020; Accepted 28 October 2020; Published 24 November 2020

Academic Editor: Liming Fang

Copyright © 2020 Xavier Salleras and Vanesa Daza. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

5G communications proposed significant improvements over 4G in terms of efficiency and security. Among these novelties, the 5G network slicing seems to have a prominent role: deploy multiple virtual network slices, each providing a different service with different needs and features. Like this, a Slice Operator (SO) ruling a specific slice may want to offer a service for users meeting some requirements. It is of paramount importance to provide a robust authentication protocol, able to ensure that users meet the requirements, providing at the same time a privacy-by-design architecture. This makes even more sense having a growing density of Internet of Things (IoT) devices exchanging private information over the network. In this paper, we improve the 5G network slicing authentication using a Self-Sovereign Identity (SSI) scheme: granting users full control over their data. We introduce an approach to allow a user to prove his right to access a specific service without leaking any information about him. Such an approach is SANS, a protocol that provides nonlinkable protection for any issued information, preventing an SO or an eavesdropper from tracking users' activity and relating it to their real identities. Furthermore, our protocol is scalable and can be taken as a framework for improving related technologies in similar scenarios, like authentication in the 5G Radio Access Network (RAN) or other wireless networks and services. Such features can be achieved using cryptographic primitives called Zero-Knowledge Proofs (ZKPs). Upon implementing our solution using a state-of-the-art ZKP library and performing several experiments, we provide benchmarks demonstrating that our approach is affordable in speed and memory consumption.

1. Introduction

5G communications enhanced the way how mobile devices are connected to cellular networks. They not solely improved the 4G Radio Access Network (RAN) but also introduced a new paradigm where devices with different specifications are routed through different physical and logical networks, called network slices. This opened new business models, for instance, creating network slices for specific services offered by third parties. Like this, a Slice Operator (SO) ruling a network slice may want to offer a service to users meeting some requirements (e.g., users enrolled in a governmental program and users who have paid for using such a service). Among the growing density of Internet of Things (IoT) devices using 5G

communications, we can find examples of devices sharing sensitive data over the network: medical devices exchanging private information or autonomous cars sharing their location with a network slice. Needless to say, this data should not be traced by any SO or eavesdropper. In such a scenario, traditional authentication schemes leak all this data to the SO. As such, Self-Sovereign Identity (SSI) [1] becomes an important feature to implement: systems where users can control, access, and transparently consent their identities, preventing entities from tracking and gathering their personal data. Likewise, the main idea behind SSI systems is to provide a unique mechanism for users to authenticate into different services, providing only the required information, information which shall be nontraceable.

1.1. Contributions. We introduce SANS, a novel self-sovereign authentication approach where a user demonstrates his right to access a service, without leaking any information about him. Our approach is an underlying protocol to be integrated into existing SSI systems, avoiding any user activity to be linked with any other activity done in the past or the future. Moreover, it also prevents the SO or an attacker impersonating him from tracking users' activity. Our protocol grants the user with these main features:

- (i) Anonymity: the SO has no way to relate any digital identity to a real identity.
- (ii) Proof of requirements: the user can prove that he meets the requirements needed for using a specific service.
- (iii) Nonlinkable activity: the SO has no way to relate any user activity to another activity done in the network.

To achieve the aforesaid key features, we use Zero-Knowledge Proofs (ZKPs), cryptographic primitives gaining a lot of momentum in the last years. Since the seminal paper in [2], demonstrating how ZKPs can prove knowledge of a secret without leaking any information about it, several applications have been envisioned. However, during decades, they were far from being used in real-life applications due to nonexistent efficient implementations. Nevertheless, recently, many efficient ZKPs have broken into the scene, revolutionizing not only the state-of-the-art in the area but also the market in scenarios like cryptocurrencies (e.g., Zcash [3]) or smart-contracts (e.g., Ethereum [4]). Using ZKPs, we can ensure self-sovereign authentication in 5G network slices, as a user would be able to prove his right to access a specific service, requested by an SO, without leaking any information about him.

1.2. Roadmap. In Section 2, we expose the background required to understand the context of the problem and also our solution. In Section 3, we introduce the relevant work done concerning our topic. In Section 4, we present our protocol with all details, including the security analysis, the implementation, and the performed experiments. Conclusions are provided in Section 5.

2. Background

In this section, we first introduce the basics of 5G network slicing, and later, for the sake of completeness, we provide an overview of what ZKPs are and how they could be applied to our protocol.

2.1. 5G Network Slicing. 5G is the fifth generation of mobile communications [5], which achieves faster speeds than LTE networks and more reliable service. The 5G network is split into different network slices, which are independent networks dedicated and optimized for specific services. This new architecture is built employing Software-Defined Networking (SDN) and Network Functions Virtualization (NFV), along with the physical infrastructure. All these

changes lead to higher performance: higher speeds, lower delays, and much less network latency. As depicted in Figure 1, different kinds of User Equipment (UE) are part of different slices, depending on their specifications or the services they are willing to use. In a nutshell, the main network slices are as follows:

- (i) eMBB slice: The enhanced Mobile Broadband (eMBB) slice is meant for services that require high bandwidth, like Internet browsing, high definition video streaming, virtual reality, and so on.
- (ii) mMTC slice: The massive Machine Type Communications (mMTC) slice aims to group a high density of devices, which do not have other essential requirements like a low latency or a high bandwidth. Examples of this are IoT devices, specifically in the context of smart cities.
- (iii) uRLLC slice: The ultra-Reliable and Low-Latency Communications (uRLLC) slice aims to provide very low network latency, a crucial requirement for services like autonomous driving or remote management.

As depicted in Figure 1, users connect their UE to the small 5G cells of the 5G RAN, which forward the connections to the 5G core network, split into different software-defined networks (i.e., eMBB, mMTC, and uRLLC).

Furthermore, access to the 5G core network is allowed not solely from the new 5G RAN but also from other networks like the 4G RAN or optical fiber connections, depending on the requirements of the service. As such, we understand 5G as a heterogeneous network (HetNet), a network interconnecting devices with different specifications and protocols, where a common and trustworthy authentication scheme would be a desirable feature.

2.2. Zero-Knowledge Proofs. A Zero-Knowledge Proof (ZKP) [2] is a cryptographic primitive which allows a prover P to convince a verifier V that a statement is true, without leaking any secret information. In particular, ZKPs must satisfy 3 properties:

- (i) Completeness: if the statement is true, P must be able to convince V .
- (ii) Soundness: if the statement is false, P must not be able to convince V that the statement is true, except with negligible probability.
- (iii) Zero-knowledge: V must not learn any information from the proof beyond the fact that the statement is true.

Moreover, V may also be interested in an additional property, the proof of knowledge, which guarantees that P knows the secret information about the statement. This secret information that the prover knows is usually called witness w . In other words, P wants to prove knowledge of a secret witness w for which a set of operations hold. Such operations are defined by a circuit, a graph composed of different wires and gates, which leads to a set of equations

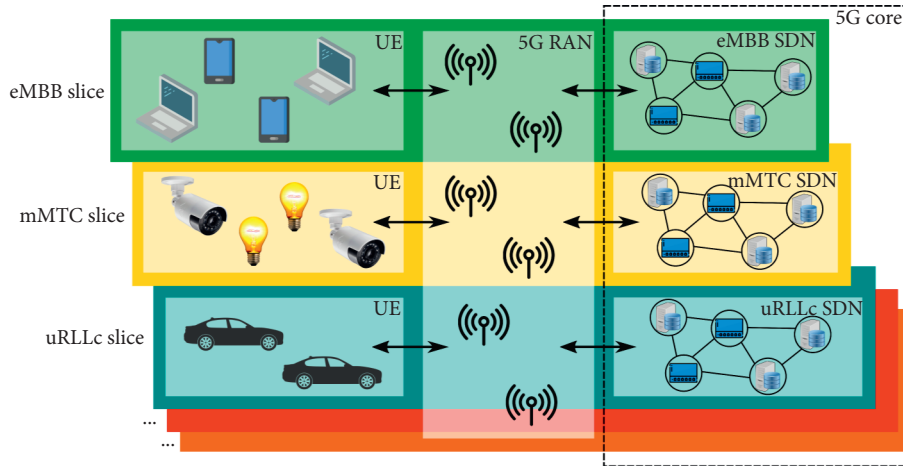


FIGURE 1: General 5G architecture overview.

relating to the inputs and the outputs of these gates. Each of these equations is called a constraint.

To achieve their goal, both P and V need to interact several times. However, as iterating is not always a desirable property, another kind of ZKPs called Noninteractive ZKP (NIZKP) [6] arose. In this case, P proves a statement to V by sending him a single message, without interaction. First NIZKP schemes were far from being implemented, due to their impractical computing requirements. Here is where one of the most popular ZKPs arose, zk-SNARKs, which are Zero-Knowledge Succinct and Noninteractive ARGuments of Knowledge [7]. This kind of proof is short and succinct: it can be verified in a few milliseconds. In this scheme, a trusted setup is required, in order to get some public parameters used by either P or V as a reference to generate and verify proofs. These parameters are called the Common Reference String (CRS). If an attacker was able to get the secret random values τ used to generate the CRS, he would be able to generate false proofs. For this reason, the initial setup is commonly made through a secure Multiparty Computation (MPC) [8], which generates the required parameters using a distributed computation protocol. Therefore, zk-SNARKs are composed of three algorithms: setup, prove, and verify. The computing complexity of some of these elements depends on the number of gates n , which is the number of operations that we do for proving a specific statement.

There are also other interesting kinds of ZKPs rather than zk-SNARKs, like Bulletproofs [9]. As shown in Table 1, Bulletproofs are constructions whose proof size is larger than zk-SNARKs, where the complexity is $O(\log n)$ versus the constant proof size complexity of zk-SNARKs. Moreover, zk-SNARKs are also faster in verifying time complexity. Even when Bulletproofs have linear proving time complexity, the large number of operations required for every constraint leads to a high proving time in practice. As such, the main advantage of Bulletproofs is that they do not require a trusted setup.

Another interesting kind of ZKPs is zk-STARKs (Zero-Knowledge Succinct Transparent ARGument of Knowledge)

[10], whose size is much higher than zk-SNARKs and Bulletproofs ($O(\log^2 n)$). One of their main advantages is that like Bulletproofs; they do not require a trusted setup. Another advantage of zk-STARKs is that they are supposed to be postquantum secure, which is not the case of zk-SNARKs and Bulletproofs.

Regarding the security of the schemes described above, the soundness property of each scheme relies on different security assumptions [11]. As shown in Table 1, zk-SNARKs use a strong assumption, the q -Power Knowledge of Exponent (q -PKE) assumption, while Bulletproofs or zk-STARKs use better approaches: the Discrete Logarithm Problem (DLP) and Collision Resistant Hash Functions (CRHF), respectively.

However, one of the most important improvements regarding ZKPs is the zk-SNARK construction introduced in [12], which introduces the most efficient zk-SNARK designed so far. One of its main improvements is that the verifier has to evaluate a single equation, using only three pairings, instead of five equations and twelve pairings, as done in [7]. Such improvements led to a huge usage of this construction in different applications like Zcash.

Another critical research topic is resilience against quantum attacks. An essential contribution regarding this topic has been done in [13, 14], where a new zk-SNARK construction believed to be postquantum secure is introduced.

Regarding the scalability of the implementations, a significant contribution has been done in [15]. They propose Sonic, a zk-SNARK construction which requires a trusted setup, but with the difference that such a setup supports different circuits and is also updatable, meaning that the scheme can be continuously improved. As during the setup, a CRS is made public, by using an updatable CRS model [16], any user can update the CRS, and he can also prove that it was done correctly, employing a proof of correctness. If this proof is verified, the new CRS can be trusted as long as either the old CRS or the user who did the update was honest. Moreover, zk-SNARK constructions without the need for a trusted setup have also been designed, like the one in [17].

TABLE 1: Comparison of different ZKP constructions, where n is the number of gates of the circuit.

	Trusted setup	Prove	Verify	Proof size	Assumption
zk-SNARKs [7]	Yes	$O(n \log n)$	$O(1)$	$O(1)$	$q - \text{PKE}$
Bulletproofs [9]	No	$O(n)$	$O(n)$	$O(\log n)$	DLP
zk-STARKs [10]	No	$O(n \log^2 n)$	$O(\log^2 n)$	$O(\log^2 n)$	CRHF

Having in mind the schemes described above, the ZKP construction that best fits our solution (at the moment of writing this) is zk-SNARKs. We need proofs to be succinctly verifiable to not overload the verifier and at the same time, it is also preferred to have proofs with a constant size. In that regard, the Groth'16 construction [12] provides a reasonably efficient prover, so it could be the preferred option for SANS, as having a construction with efficient proving and verifying algorithms is of paramount importance in our scenario. In Section 4.4, we show the results of several experiments we have done in this regard. However, we recall the fact that our solution could be used with other ZKP constructions if better options arise.

3. Related Work

Self-Sovereign Identity (SSI) has gained a lot of interest in the last years. The author in [18] envisioned an SSI system where users can control, consent, and widely use their identities among different services, along with other properties. These properties were redefined in [1] by the Sovrin Foundation (<https://sovrin.org/>). They introduced the guidelines on how SSI systems can be implemented along with blockchain technologies, providing a distributed architecture of trust without central authorities managing users' data. In this regard, SSI authentication schemes like the one proposed in [19] make use of blockchain technologies for deploying a decentralized and private authentication system.

A good review of the state-of-the-art regarding this topic is done in [20]. As they state, ZKPs allow a user to prove ownership of an identity, that is, proving knowledge of a secret key related to a public key stored in a blockchain.

As stated previously, the core of network slicing relies on an SDN-based architecture. In this regard, interesting research is addressed in [21], where a novel authentication scheme preventing multiple types of SDN authentication attacks is introduced. This makes even more sense in the context of a medical cloud sharing sensitive information, a fact that has led to schemes [22] guaranteeing a secure authentication in this scenario.

A more specific use case related to our approach is introduced in [23]. They state some of the benefits of SSI for IoT devices, like the fact that the identities of the owners of different devices are stored locally in the devices, rather than on a centralized entity (i.e., the SO in our scenario). As explained by the authors, SSI provides a layered authentication system separating application authentication from channel authentication, where the former handles the trust requirements. This grants a more reliable end-to-end security, where secure communication is established among different protocols.

Among the aforesaid studies regarding SSI, to the best of our knowledge, there are no solutions applied to 5G network slices. In this regard, we propose a solution to integrate SSI into network slices in the next section.

4. Our Solution: SANS

In this section, we first explain our approach with all the required details. Later, we analyze the security of our protocol, its computing constraints, and its benchmarks.

4.1. Protocol Description. We start with a high-level description of SANS and later move to a more detailed one: a user willing to join a network slice to use its service may be required to meet some requirements, like having paid a subscription fee. As such, the user is a prover P willing to prove to a verifier V , the SO, that he has paid such an amount (the statement). Our protocol accomplishes this purpose. To do so, an important requirement of our protocol is being able to prove knowledge of contracts signed using a given secret key: P must convince V that he knows a contract and its signature, which is verified using a public key. The contract can be a secret value, and still, V must be convinced. In order to be efficient, the used signing algorithms have to be ZKP-friendly, and this means that its operations can be reduced to a low number of constraints. For instance, the Edwards-curve Digital Signature Algorithm (EdDSA) [24] is a fast signing algorithm widely used with zk-SNARKs. Moreover, signature algorithms in zk-SNARKs must be combined with efficient hashing functions as well. One of the most efficient zk-SNARK-friendly hashes to the date is Poseidon [25], which needs 8 times fewer constraints for its circuit than the widely used Pedersen hash.

Our authentication scheme is divided into two protocols, depicted altogether in Figure 2. The first one is the service registration protocol, to be performed for each issued payment. Its steps are as follows.

Protocol 1. Service registration

- (1) P provides V some requested information req (e.g., a statement from the bank stating that a payment has been issued).
- (2) After verifying req, V generates a unique byte-array token identifying the user and sends it to him along with a timestamp t_{exp} representing the contract expiration date. Moreover, V provides a signature $S = \text{sign}(\text{token}, t_{\text{exp}})$ and its public key pk_{SO} .

After having registered into the service, the user can use the provided parameters to authenticate into the service each time it needs to use it and thus create a new session into the

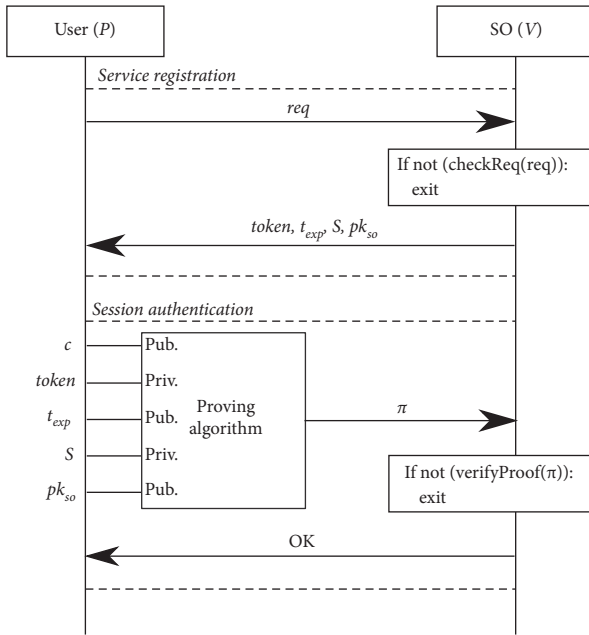


FIGURE 2: Overview of the service registration and session authentication.

service. Moreover, in order to avoid replay attacks [26] (i.e., an eavesdropper taking the proof and replying it to the SO), every proof must include the hash of the secret token concatenated to a variable public parameter c . Further details of such an approach are discussed in Section 4.2. The session authentication protocol is performed as follows.

Protocol 2. Session authentication

- (1) P computes a proof π whose circuit inputs are

- c (public input);
- $token$ (private input);
- S (private input);
- pk_{SO} (public input);
- t_{exp} (public input).

- (2) V verifies the proof π and grants the service.

The generated proof π proves knowledge of the secret inputs of the circuit depicted in Figure 3. As shown, we prove knowledge of the hash of a secret token concatenated to its expiration date t_{exp} . This is our contract, and we also prove that we know its secret signature (signed by V) using the public key pk_{SO} . This outputs 0/1 if the signature is verified or not, respectively, and this value is multiplied by the output of hash($c, token$). If all is correct, the circuit will output a hash; otherwise, the output will be 0.

4.2. Security Analysis. In this section, we analyze the security of our solution. We also detail how to overcome some possible attacks.

4.2.1. False Proofs Generation. The main drawback of some ZKP constructions like zk-SNARKs is the need for a trusted

setup. In many scenarios, like in Zcash, an untrusted setup could lead to huge losses of money if a malicious party gets the trapdoor τ and starts to create false transactions. However, this is not a problem in our solution: a different setup can be generated by each SO. If the SO keeps and spreads the trapdoor τ , anyone knowing τ will be able to access the service by generating false proofs. As such, the protocol is secure as long as the setup is generated only by the SO and he destroys τ . Furthermore, as stated previously, the ZKP construction that best fits our solution at the moment of writing this is the Groth'16 zk-SNARK. As such, the security of SANS depends on a q -PKE assumption.

4.2.2. Elliptic Curve Attacks. The security of our solution also relies on the security of elliptic curves. One of the most used curves in ZKPs is a Barreto-Naehrig curve [27] called BN128, of which the security level in practice is estimated to be 110 bits [28]. This means that an attacker willing to break BN128 shall perform 2^{110} operations. Other curves like BLS12-381 [3] estimate around 128 bits of security, with the drawback of heavier group operations. Breaking the security of the used elliptic curve would lead to being able to generate false proofs.

4.2.3. Account Sharing. Every computed proof is different since it is generated using random parameters, allowing the user to generate different proofs with the same inputs. As such, the user could generate multiple proofs for other users, which would access the service with a single subscription. To overcome this issue, a simple solution is integrated into our protocol: every proof must include the hash of the secret token concatenated to a variable public parameter c . Ideally, this parameter could be a timestamp with a specific accuracy, for instance, the date in format yyyy/mm/dd plus the time in format hh:mm without seconds. Such a hash should be multiplied by the output of the verification of the signature (1 or 0, if verified or not, resp.), and if everything is correct, the circuit should output a hash. Like this, an SO receiving the same hash more than once could identify that those proofs have been computed using the same token. As such, if two users are trying to use the service at the very same time, the SO can relate and reject both connections.

4.2.4. 5G RAN Authentication. One of the main concerns about our solution is to provide a fully private authentication, where the SO cannot learn the identity of the user. In this scenario, we still have another party, the Internet Service Provider (ISP), who acts as an SDN controller providing the architecture and the workflows for optimal network slicing. As such, the ISP learns the identities of the users from the moment that the UE accesses the 5G RAN. To overcome this, we envision the usage of SANS when the UE is required to authenticate for accessing the 5G RAN. In other words, the UE would be proving his right to access the 5G RAN, for instance, by proving that the user has paid the last month bill to the ISP.

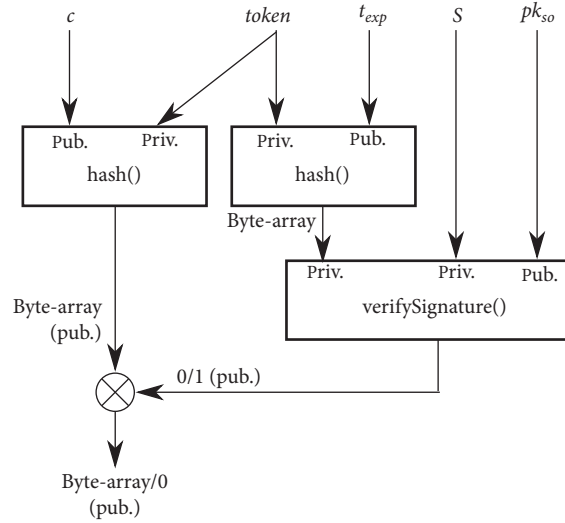


FIGURE 3: Overview of the circuit used by the session authentication protocol.

4.3. *Efficiency Analysis.* This section describes several efficiency considerations of SANS.

4.3.1. *Computational Complexity.* As we saw in Section 2, the setup protocol depends only on the number of gates, so this protocol has a linear computing complexity $O(n)$. The most consuming operation done by the prover is to compute the coefficients of a polynomial $H(x)$, which can be computed more efficiently employing Fast Fourier Transform (FFT) techniques [29], leading to a computing complexity of $O(n \log n)$. The verifier has to do a constant computation of group exponentiations and an equation composed of three pairings.

4.3.2. *Prover Optimizations.* There are different operations performed by the zk-SNARK prover which can be parallelized in order to improve its efficiency. This means that CPU and GPU multiprocessing techniques can be applied to speed up the implementations. Even so, the usage of external computing resources as done in [30] can be taken into account. For instance, in the case of a prover being a smartwatch with low computing resources, the heaviest computations could be precomputed by the user's phone, whose computing power should be higher.

4.3.3. *Circuit Size.* Our circuit contains a single EdDSA signature combined with two hashes (to the date of writing this, Poseidon seems to be the best option). The authors of circomlib (<https://github.com/iden3/circomlib/>) developed optimal EdDSA and Poseidon circuits, which leads our solution to a total size of 7565 constraints and affordable computing times as shown in the next subsection.

4.4. *Implementation and Benchmarks.* We implemented (<https://github.com/xavisalle/sans>) our solution using snarkjs, a JavaScript and WASM framework for

implementing zk-SNARK applications. The reason for choosing this option is its simplicity for implementing circuits and its portability in web environments. In this regard, we deployed our implementation in a web server, to be executed by different devices using different web browsers. Overall, the number of constraints of this implementation is 7565, and as depicted in the chart of Figure 4, our solution outperforms in high-performance CPUs (i7-8750H), using either Mozilla Firefox or Google Chrome. As such, our solution could be used in desktop applications with no problems with regard to performance.

On the other hand, the proving time increases notably in low-performance processors (Intel Atom x7-Z8750), achieving timings higher than 2 seconds in both Firefox and Chrome. An interesting fact is how Chrome performs slightly better than Firefox in its desktop version, which does not apply to mobile CPUs (Snapdragon 845). Regarding Snapdragon 845, even when it is a top mobile processor, we can see that the results are not as good as i7-8750H. However, the achieved results prove that our solution is feasible in performance, especially when the portability is a priority. Moreover, the memory consumption has been in all tests between 150 and 200 MB (not taking into account what is consumed by default by the browsers).

Furthermore, we also tested libsnark (<https://github.com/scipr-lab/libsnark>), a well-optimized C++ zk-SNARKs library achieving excellent benchmarks, but with the drawback of not being as portable as other solutions like snarkjs. For instance, as the authors of libsnark state, the library is not well-optimized for ARM architectures (e.g. Snapdragon 845), and the BN128 curve is not supported in this architecture.

We implemented a circuit with the same amount of constraints that our solution has, and we executed the prover in multicore mode using Groth'16 and the BN128 curve. The obtained results are shown in the chart of Figure 5. As can be seen, libsnark achieves much better results than snarkjs, so implementing SANS using this library would be even more

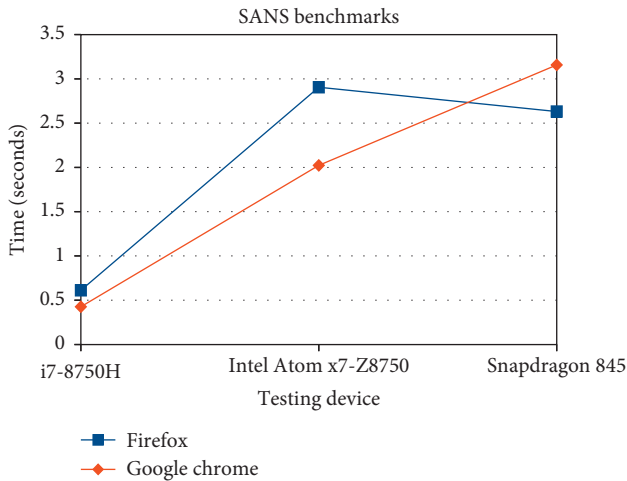


FIGURE 4: CPUs proving times comparison of SANS using a snarkjs implementation executed in multicore mode for browsers. The used zk-SNARK is Groth'16, and the elliptic curve BN128. Intel Atom x7-Z8750 and i7-8750H run desktop browsers for Linux; Snapdragon 845 runs Firefox and Chrome for Android 10.

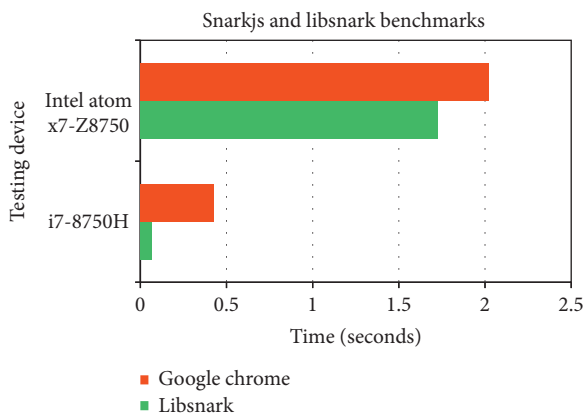


FIGURE 5: CPUs proving times comparison of snarkjs and libsark 7565 constraint circuits executed in multicore mode. The used zk-SNARK is Groth'16, and the elliptic curve is BN128. Intel Atom x7-Z8750 and i7-8750H run a desktop Linux distribution.

feasible. Regarding the memory consumption, libsark performs better as well: around 20 MB in both tested devices. Furthermore, optimized libraries for mobiles and embedded systems would lead to additional performance improvement, so future work in this regard would be an exciting research topic.

5. Conclusions

In this paper, we have introduced SANS, a protocol for proving the right of a user to access a specific 5G network slice, without leaking any information about him beyond the fact that he possesses such a right. Our solution is an underlying protocol to be integrated into existing SSI schemes. Moreover, it could be easily extended to other scenarios, like 5G RAN authentication, other kinds of wireless communications, or distributed applications. Even when some ZKP

schemes like zk-SNARKs require costly computing operations, we have proved our solution to be affordable in terms of efficiency and memory consumption by implementing SANS using existing libraries. Furthermore, we proved the portability of our implementation by testing it on several devices. Nevertheless, future work on optimized ZKP libraries for embedded systems would be interesting, to spread the usage of this protocol.

Data Availability

All the data are included in the article itself.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors were supported by Project RTI2018-102112-B-100 (AEI/FEDER, UE).

References

- [1] Sovrin Foundation. Sovrin: A Protocol and Token for Self-Sovereign Identity and Decentralized Trust. <https://sovrin.org/wp-content/uploads/Sovrin-Protocol-and-Token-White-Paper.pdf>, January 2018.
- [2] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems," in *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, STOC '85*, pp. 291–304, ACM, New York, NY, USA, December 1985.
- [3] Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. Zcash Protocol Specification - Version 2019.0.2, 2019. <https://github.com/zcash/zips/blob/master/protocol/protocol.pdf>.
- [4] B. Bünz, S. Agrawal, M. Zamani, and D. Boneh, "Zether: towards privacy in a smart contract world," *Cryptology ePrint Archive, Report 2019/191*, <https://eprint.iacr.org/2019/191>, 2019.
- [5] ETSI (3GPP). Procedures for the 5G System (5GS), v15.5.1, release 15, May 2019.
- [6] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, STOC '88*, pp. 103–112, ACM, New York, NY, USA, January 1988.
- [7] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von neumann architecture," *Cryptology ePrint Archive, Report 2013/879*, 2013, <https://eprint.iacr.org/2013/879>.
- [8] S. Bowe, A. Gabizon, and I. Miers, "Scalable multi-party computation for zk-SNARK parameters in the random beacon model," *Cryptology ePrint Archive, Report 2017/1050*, 2017, <https://eprint.iacr.org/2017/1050>.
- [9] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: short proofs for confidential transactions and more," *Cryptology ePrint Archive, Report 2017/1066*, 2017, <https://eprint.iacr.org/2017/1066>.
- [10] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure

- computational integrity,” Cryptology ePrint Archive, Report 2018/046 <https://eprint.iacr.org/2018/046>, 2018.
- [11] S. Goldwasser and Y. T. Kalai, “Cryptographic assumptions: a position paper,” in *Theory of Cryptography*, E. Kushilevitz and T. Malkin, Eds., pp. 505–522, Springer, Berlin, Heidelberg, 2016.
- [12] J. Groth, “On the size of pairing-based non-interactive arguments,” Cryptology ePrint Archive, Report 2016/260, 2016, <https://eprint.iacr.org/2016/260>.
- [13] R. Gennaro, M. Minelli, A. Nitulescu, and M. Orrù, “Lattice-based zk-snarks from square span programs,” Cryptology ePrint Archive, Report 2018/275, 2018, <https://eprint.iacr.org/2018/275>.
- [14] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward, “Aurora: transparent succinct arguments for R1CS,” Cryptology ePrint Archive, Report 2018/828, 2018, <https://eprint.iacr.org/2018/828>.
- [15] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, “Sonic: zero-knowledge SNARKs from linear-size universal and updateable structured reference strings,” Cryptology ePrint Archive, Report 2019/099, 2019, <https://eprint.iacr.org/2019/099>.
- [16] J. Groth, M. Kohlweiss, M. Maller, S. Meiklejohn, and I. Miers, “Updatable and universal common reference strings with applications to zk-SNARKs,” Cryptology ePrint Archive, Report 2018/280, 2018, <https://eprint.iacr.org/2018/280>.
- [17] R. S. Wahby, I. Tzialla, A. Shelat, J. Thaler, and M. Walfish, “Doubly-efficient zkSNARKs without trusted setup,” Cryptology ePrint Archive, Report 2017/1132, 2017, <https://eprint.iacr.org/2017/1132>.
- [18] Christopher Allen. The path to self-sovereign identity. Accessed 2020-07-17. <http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html>.
- [19] A. Othman and J. Callahan, “The horcrux protocol: a method for decentralized biometric-based self-sovereign identity,” in *Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, Rio de Janeiro, Brazil, July 2018.
- [20] M. Alexander, A. Grüner, T. Gayvoronskaya, and C. Meinel, “A survey on essential components of a self-sovereign identity,” *Computer Science Review*, vol. 30, pp. 80–86, 2018.
- [21] L. Fang, Y. Li, X. Yun et al., “THP: a novel authentication scheme to prevent multiple attacks in SDN-based IoT network,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5745–5759, 2020.
- [22] L. Fang, C. Yin, L. Zhou, Y. Li, C. Su, and J. Xia, “A physiological and behavioral feature authentication scheme for medical cloud based on fuzzy-rough core vector machine,” *Information Sciences*, vol. 507, pp. 143–160, 2020.
- [23] G. Fedrecheski, J. M. Rabaey, L. C. D. P. Costa, W. T. Pereira, and M. K. Zuffo, “Self-sovereign identity for IoT environments: a perspective,” in *Proceedings of the 2020 Global Internet of Things Summit (GIoTS)*, pp. 1–6, University of São Paulo, São Paulo, Brazil, March 2020.
- [24] D. J. Bernstein, N. Duif, T. Lange, S. Peter, and B.-Y. Yang, “High-speed high-security signatures,” *Journal of Cryptographic Engineering*, vol. 2, 2012, <https://cr.yp.to/papers.html#ed25519>.
- [25] L. Grassi, D. Khovratovich, C. Rechberger, A. Roy, and M. Schafneger, “Starkad and poseidon: new hash functions for zero knowledge proof systems,” Cryptology ePrint Archive, Report 2019/458, 2019, <https://eprint.iacr.org/2019/458>.
- [26] V. Daza and X. Salleras, “LASER: lightweight And SEcure Remote keyless entry protocol (Extended version),” 2019, <https://arxiv.org/pdf/1905.05694.pdf>.
- [27] P. S. L. M. Barreto and M. Naehrig, “Pairing-friendly elliptic curves of prime order,” Cryptology ePrint Archive, Report 2005/133, 2005, <https://eprint.iacr.org/2005/133>.
- [28] A. Menezes, P. Sarkar, and S. Singh, “Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography,” Cryptology ePrint Archive, Report 2016/1102, 2016, <https://eprint.iacr.org/2016/1102>.
- [29] V. Migliore, M. M. Real, V. Lapotre, A. Tisserand, C. Fontaine, and G. Gogniat, “Exploration of polynomial multiplication algorithms for homomorphic encryption schemes,” in *Proceedings of the 2015 International Conference on ReConfigurable Computing and FPGAs (ReConFig)*, pp. 1–6, Riviera Maya, Mexico, December 2015.
- [30] H. Wu, W. Zheng, A. Chiesa, R. A. Popa, and I. Stoica, “DIZK: a distributed zero knowledge proof system,” Cryptology ePrint Archive, Report 2018/691, 2018, <https://eprint.iacr.org/2018/691>.