

Emerging Authentication, Identification, and Authorization Technologies

Lead Guest Editor: David Chadwick

Guest Editors: Philippe Palanque, Romain Laborde, Omar Alfandi, and Ahmad Samer Wazan





Emerging Authentication, Identification, and Authorization Technologies

Security and Communication Networks

Emerging Authentication, Identification, and Authorization Technologies

Lead Guest Editor: David Chadwick

Guest Editors: Philippe Palanque, Romain Laborde,
Omar Alfandi, and Ahmad Samer Wazan






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

Smart Identity Management System by Face Detection Using Multitasking Convolution Network

Lubna Farhi , Hira Abbasi , and Rija Rehman 


Research Article (11 pages), Article ID 7314823, Volume 2021 (2021)

Linkable Ring Signature Scheme Using Biometric Cryptosystem and NIZK and Its Application

Xuechun Mao , Lin You , Chengtang Cao , Gengran Hu , and Liqin Hu 





Research Article (14 pages), Article ID 7266564, Volume 2021 (2021)

Security Analysis and Bypass User Authentication Bound to Device of Windows Hello in the Wild

Ejin Kim  and Hyoung-Kee Choi 


Research Article (13 pages), Article ID 6245306, Volume 2021 (2021)

An Efficient User-Centric Consent Management Design for Multiservices Platforms

Paul Marillonnet , Mikaël Ates , Maryline Laurent , and Nesrine Kaaniche 

Research Article (19 pages), Article ID 5512075, Volume 2021 (2021)

Privacy-Preserving Two-Factor Key Agreement Protocol Based on Chebyshev Polynomials

Zuowen Tan 

Research Article (21 pages), Article ID 6697898, Volume 2021 (2021)

Research Article

Smart Identity Management System by Face Detection Using Multitasking Convolution Network

Lubna Farhi , Hira Abbasi , and Rija Rehman 

Department of Electronic Engineering, Sir Syed University of Engineering and Technology, Karachi, Pakistan

Correspondence should be addressed to Lubna Farhi; lubnafarhi@yahoo.com

Received 25 June 2021; Revised 31 October 2021; Accepted 24 November 2021; Published 21 December 2021

Academic Editor: Ahmad Samer Wazan

Copyright © 2021 Lubna Farhi 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.

Identity management system in most academic and office environments is presently achieved primarily by a manual method where the user has to input their attendance into the system. The manual method sometimes results in human error and makes the process less efficient and time-consuming. The proposed system highlights the implementation and design of a smart face identification-based management system while taking into account both the background luminosity and distance. This system detects and recognizes the person and marks their attendance with the timestamp. In this methodology, the face is initially resized to 3 different sizes of 256, 384, and 512 pixels for multiscale testing. The overall outcome size descriptor is the overall mean for these characteristic vectors, and the deep convolution neural network calculates 22 facial features in 128 distinct embeddings in 22-deep network layers. The pose of the 2D face from -15 to $+15^\circ$ provides identification with 98% accuracy in low computation time. Another feature of the proposed system is that it is able to accurately perform identification with an accuracy of 99.92% from a distance of 5 m under optimal light conditions. The accuracy is also dependent on the light intensity where it varies from 96% to 99% under 100 to 1000 lumen/m², respectively. The presented model not only improves accuracy and identity under realistic conditions but also reduces computation time.

1. Introduction

In many public and educational sectors, the management system is mandatory for analyzing the performance of candidates. When there are a lot of individuals in an organization or institute, it becomes significantly more difficult to mark their presence through the manual procedure and it is also time-consuming. The conventional marking method is obsolete, and in such systems, identification is recorded with traditional approaches that include registers and sheets whereas more advanced methods like RFID and biometric encounter the difficulty of time wastage and are significantly more complicated where you have to wait in line to swipe the RFID card or put your thumb on a scanner which can be a quick way of spreading unwanted diseases. It is also prone to manipulation where individuals can mark the presence of others without any oversight if they possess the RFID card. Sorting and calculating the attendance of every enrolled person is not only tiresome but humans can easily make

mistakes while conducting repetitive tasks. Therefore, a smart system is required for marking and recording. In order to do that, we will save an authentic and proper record of persons that can also be analyzed later on if needed.

In addition to reducing errors, the proposed system for management is also more feasible than other methods. For example, the biometric system needs more hardware, and its maintenance is also difficult. The automatic system can resolve a crucial issue within the manual one that occurs when a person transfers the information from the sheet into the system. The face identification method has many steps which include capture, extraction, comparison, and matchmaking. An automated and computerized attendance information and management system with enhanced face identification has been proposed. The initial steps include database creation, face identification, feature engineering, and categorization stages followed by the last stage, i.e., postprocessing phase [1]. At first, facial images of every student would be transferred to the system and saved within

the database. Then, the identification of candidates is recorded by using a camera attached within the area at an appropriate location from where the entire region is often viewed or monitored. The camera will constantly take pictures of candidates, identify the countenances in pictures, recognize the identified countenances, and mark their identity. In some methods, the camera is at a fixed position at the point of entry to capture the image of the candidate as they enter that area. Through this technique, we will save more time as compared to the manual management system. Finally, if sorting is needed, then it can also be done easily.

There have been many types of research work on surveillance systems that have been done on various devices; most of them were embedded systems based on GPU as well as FPGA [2]. The most effective and powerful GPUs [3] have been utilized to implement the rules of the monitoring mechanism for standard facial identification and object identification algorithms [4] with an accuracy that was nearly equal to 88 percent [5]. However, in recent years, various deep learning architectures have shown optimistic accuracy. This includes FaceNet [6] which provides nearly 99 percent on a system based on GPU. Face recognition has been achieved through various approaches. Some of them are feature-based, multimodal fusions, holistic appearance, or multispectral-based implemented for face recognition in the infrared spectrum. Early research on infrared imagery for facial recognition was carried out with the introduction of the method based on eigenfaces. They produced a recognition rate of approximately 96% by running a database of 24 subjects with 12 images, each forming a database of 288 thermal images [7]. The base images in this technique represented the variation of posture and face. Different types of improvements in the linear methods, such as eigenfaces, Local Feature Analysis (LFA), Independent Component Analysis (ICA), and Linear Discrimination Analysis (LDA), significantly improved the precision of the images in thermal and visible data-based face recognition. More specifically, the increase in precision was much higher in the thermal spectrum (about 93% to 98%) than in the visible spectrum. However, despite the improvements, harmful variables persisted in the image databases that have the potential for increasing bias and skewing the result. Similarly, holistic appearance approaches take into account the picture of the face. Such a methodology is unique in the way facial images work and helps to treat faces in a different way from other categories discussed in [8]; therefore, it is not responsible for the independent processing of functions. This approach has been used by various researchers in facial recognition using infrared imagery. Some have investigated the potential of infrared imagery for facial recognition by extraction of a significant shape called “elemental forms,” and the structure of these elemental forms was similar to fingerprints [9]. A methodology built on a general Gaussian mixture model uses the Bayesian approach to select the parameters from a sample image [10]. However, this research resulted in an approximately 95% facial identification accuracy in the thermal and appearance-based data. Another facial recognition approach was created with a database of 50 people along with 10 images per individual, which offered authentic

evidence for facial recognition within the IR spectrum [11]. In this research, the data considered for classification did not include varieties of intrapersonal variables which is due to different emotional states, or exercise, or even the temperature of the air which is a major drawback in this set of research. The classification approach depends on the results based on a combination of neural networks and local averted appearances and extracts the characteristics of thermal images proposed [12]. The approach was carried out at an ambient temperature from 302 K to 285 K and achieved a recognition rate of 95% when the test and training data were entered at an identical ambient temperature. Over this period, the approach achieved the closest rate of 60 percent for recognition when the difference between training and sample data temperature remained at 17 K. The multimodal method approaches transform coded greyscale projections, eigenfaces, and pursuit filters for matching the pictures in the research of [13]. One depends on the level of data and the other on the decision-making lever. In this method, characteristics are built up by inheriting the data from the two modalities and then further classified. While within the decision level, the precision of two-person matching within the ROI and visible spectrum is computed, which makes the model more complex. Another problem in face recognition is time-lapse; i.e., the performance of an algorithm decreases as time passes between training and test data without taking into account the scanning conditions. Similarly, using the effects of atmospheric temperature on facial temperature and improving the image to standardize the facial regions is studied in [14, 15]. It shows that facial recognition errors produced in the visible spectrum and infrared spectra are affected by the passage of time between sampling and the acquisition of the test data. Later on, it had been observed that face recognition performance decreases due to changes in certain tangible factors that have an impact on the appearance of the face and in particular on the thermal data [16].

The Convolutional Neural Network (CNN) consists of a combination of convolutional layers, nonlinear layers (e.g., mean, max, or min), and classification layer units. In some cases, this methodology can be used to identify a category of a dog class, a car model, or birds, resulting in these structures having advanced potential outcomes [17]. Nowadays, most researchers are using the *Multitask Cascaded Convolution Neural Networks (MTCNN) algorithm* for facial detection and classification due to its robust nature [18, 19]. Similarly, some existing techniques and their weaknesses are discussed in Table 1.

Facial recognition approaches are hindered by many exigent challenges that include opacity of obstructions between the camera and the subject, the environmental light intensity levels, surrounding atmospheric conditions, the distance between camera and subject, and lastly but not limited to the emotional and physical expression of the subject. Moreover, most appearance-based methods supplement their analysis with complex statistical techniques only to provide specific insight into the analysis instead of a holistic understanding of the outcomes. Present research models fail to incorporate multiple aforementioned factors into their studies and often aim to optimize their models

TABLE 1: Existing techniques and their limitations.

Author	Title	Model	Weaknesses
Faruk et al. [20]	“Image to Bengali Caption Generation Using Deep CNN and Bidirectional Gated Recurrent Unit”	Deep Face	When the architecture of CNN was made hybrid with multifarious detectors, it limited the variant part of an object in remarkable identification systems. However, this methodology can only be used to identify a few categories of pictures.
Alimuin et al. [21]	“Deep hypersphere embedding for real-time face recognition”	Hybrid combination of MTCNN and FaceNet	Targets only an aspect of constraints in facial features and face recognition with low accuracy.
Jin et al. [22]	“Face recognition based on MTCNN and FaceNet”	Framework of deep cascading with Neural Network	Fails to consider surrounding intensity levels and distance between camera and face.
Xiao et al. [23]	“The Improvement of MTCNN ALGORITHM: Face Recognition with Mask”	MTCNN	This model does not consider the environmental light intensity levels, surrounding atmospheric conditions, and the distance between camera and subject.

based on one variable such as improving accuracy based on surrounding light intensity levels or distance between camera and subject, but not both. As a result, their models are only accurate under specific conditions and are not pragmatic as they fail to include the interrelated factors between these variables.

Keeping these shortcomings in consideration, this study provides a novel approach where surrounding light intensity levels, angle of the facial image acquired, and distance between camera and subject are incorporated in the design of the model. This model is then optimized to not only improve accuracy under realistic conditions but also reduce computation time through postprocessing, feature extraction, and Multitasking Cascaded Convolution Neural Networks (MTCNN) algorithm.

This paper is organized into five sections. Section 1 provides an introduction and related work. Section 2 highlights the mathematical modules. Section 3 discusses the implementation of the proposed management system and experimental results in which the performance of the proposed algorithm is evaluated and the results are shown. Section 4 presents the conclusion.

2. Methodology

There are four main modules to this proposed system that are as follows: detecting a face from a real-time stream, extracting countenance, recognizing the face, and providing the countenances.

2.1. Dataset Creation. This first and foremost step to creating a self-based face recognition dataset for an in-house facial recognition system is that physical access to specific individuals is needed to collect sample images of about 126 faces. It would be a typical system for schools, companies, or other organizations where people are physically present themselves on a daily basis. To gather footage of these individuals, we can perform two methods:

- (1) Escort them into a special room where a camera is installed. Taking pictures of the person from

different angles stores the picture of that person in a labeled directory.

- (2) Implement different systems for the different rooms.

In this proposed system, the dataset creation process is going to be implemented at the time of registration of a new student on campus. 10–20 pictures of every student will be taken on-site while creating a brand-new directory for students with reference to their department batch and section and storing the images of the students inside it. At the time of training, we have to settle on a section-based training method where the encoding of every student of the respective section is stored. The dataset contains the three subsequent directories: database directory contains the whole database of the system; i.e., persons, timetables, and student information are also available as attendance records. Next, encodings contain all the encoding files. Similarly, the models have the model file of the system and results; this subdirectory contains all the face recognition testing results such as pictures and videos that contain Labeled Faces, specifically used in testing at the time of training.

2.2. Image Acquisition and Preprocessing. Acquiring images is the first key step in the face recognition method and it is the main phase of any vision system. After the image has been acquired, different strategies for handling are often applied to the image to play out the varied vision assignments required today. In any case, within the event that the image is not recognized, the planned targets might not be achievable even with the guide of any sort of image enhancement. After the image acquisition, the captured frame is passed through the multitasked cascaded convolution neural network which reduces the unwanted features and returns a cropped face. The algorithms applied to normalize the cropped face further are as follows [24, 25]:

$$X_{\text{Normalized}} = \frac{X_{\text{mean}} - X_{\text{minimum}}}{(X_{\text{maximum}} - X_{\text{minimum}})} \cdot \quad (1)$$

The mean of the cropped face feature vector is taken which is further subtracted by the minimum of the cropped

face feature vector. The result is then divided by the range of the cropped face feature vector to produce the final result. The resultant normalized face is finally resized to 160×160 . Finally, the key points and the bounding box are placed on the original image. Figure 1 represents the flow of the input frame to the detected face.

2.3. Feature Extraction. FaceNet is used as the beginning of the technique for facial recognition [6], identifying, checking, and grouping neural networks for the system. This pretrained FaceNet model contributes as a network that is associated with a layered batch and an extremely deep convolution neural network. The deep convolution neural network is supported by L2 normalization where the integration of the face is the outcome of that standardization. The face embedding is carried out during training with a triplet loss. In case the characteristics are alike, then the loss of triplet will have the lowest distance between good and bad facial points. FaceNet consists of twenty-two deep network layers, whose output is trained over these deep layers directly to achieve a facial feature in 128 distinct embeddings. Once rectified, the completely connected seam will serve as a size descriptor which is converted into a descriptor based on commonality using the embedding module to prepare a distinct feature vector from a given template. The maximum operator has been applied to these features. For specific facial recognition, classification, and verification tasks, the network needs to be refined to anticipate a significant boost. Figure 2 represents the FaceNet architecture.

2.4. Face Detection and Reduction. The detection of facial features from a provided image is a critical task when it comes to facial identification. Without having pictures of variant faces, work will not proceed. An MTCNN is used to identify and bring actual face parts from a given picture in a position to beat multifarious face recognition standards offering real-time performance with high precision studied [26]. In this system, a pretrained model (MTCNN) is used to find the candidate's face in part of the image and interpret it into greater feature facial descriptors [27].

2.4.1. Face Judgement. The initial model resizes the picture to a special degree of size in such order which gradually increases from 12×12 to 256×256 , and it is known as a picture pyramid. The subsequent facial portions are presented by the key network, called the proposal network.

The learner target may be a bipartite issue for each sample x_i which uses the cross-entropy loss function:

$$L_i^{\text{det}} = -(y_i^{\text{det}} \log(p_i) + (1 - y_i^{\text{det}})(1 - \log(p_i))), \quad (2)$$

where the probability of face sample x_i is represented by p_i which is predicted by the MTCNN. y_i^{det} stands for ground-truth; $y_i^{\text{det}} \in \{0, 1\}$ [26].

2.4.2. Enhancing Image Qualities. R-Net or refine network sharpens limiting boxes. For the applied window, the offset

(such as the width, the height, and top-left coordinate) between them and the nearest earthly truth is predicted. The loss function is the square loss function:

$$L_i^{\text{box}} = \|y_i^{\text{box}} - y_i^{\text{box}}\|_2^2, \quad (3)$$

where the regressed target y_i^{box} is the ground-truth 4-dimensional coordinate, including the width, the top-left coordinate, and the height. The R-Net property contains many types of information tagged with relevance, such as expression, blur, invalid, illumination, pose, and occlusion.

2.4.3. Feature Location. O-Net or output network serves as the final network which determines facial landmarks from the given image which is alike to bounding box regression. The loss function is as follows:

$$L_i^{\text{landmark}} = \|y_i^{\text{landmark}} - y_i^{\text{landmark}}\|_2^2, \quad (4)$$

Likewise, the regressed feature coordinate from the network is represented as y_i^{landmark} . The ground-truth contains five coordinates: two corners of the mouth, two eyes, and the nose represented by y_i^{landmark} .

As the dataset for training is different for disparate tasks over the course of training, during training, the loss of another task's training should be zero. Thus, the combination loss function should be as follows:

$$\min \sum_{i=1}^n \sum_{j \in \{\text{det}, \text{box}, l\}} \alpha_j^i \beta_j^i L_i^j, \quad (5)$$

where the amount of the training samples is represented by n and the significance of each task is α_j . In P-Net and R-Net, $\alpha_{\text{det}} = 1$, $\alpha_{\text{box}} = 0.5$, and $\alpha_{\text{landmark}} = 0.5$, while in O-Net, for gaining high accuracy face coordinates, the parameters are $\alpha_{\text{det}} = 1$, $\alpha_{\text{box}} = 0.5$, and $\alpha_{\text{landmark}} = 0.5$. β_j^i is the sample type indicator.

These networks can do facial recognition, bounding box regression, and facial landmark tracking which is why they are known as multitasking networks. These networks are cascaded as a result where various stages are taken into account with additional processing. NMS is applied in MTCNN and is used to refine the boundaries of the applicant by refine network and output network prior to delivering output. This facial detection method has numerous advantages over various poses, visual variations, and the lighting conditions of the face. Figure 3 shows the output result of the picture passed from MTCNN.

2.5. Face Recognition. An identification approach is utilized to make sure the candidate faces the task of classification with a Support Vector Machine (SVM) categorization-related problem since it was highlighted. Matching or classification tasks of a refinement problem are solved by a Support Vector Machine [28]. It increases the boundary between the classes within given input-target entries. The classifier is the result of a specific level of robustness at overfeeding. The range represents the effectiveness of class separation. SVM finds the optimal separation of closest points in the training set. This

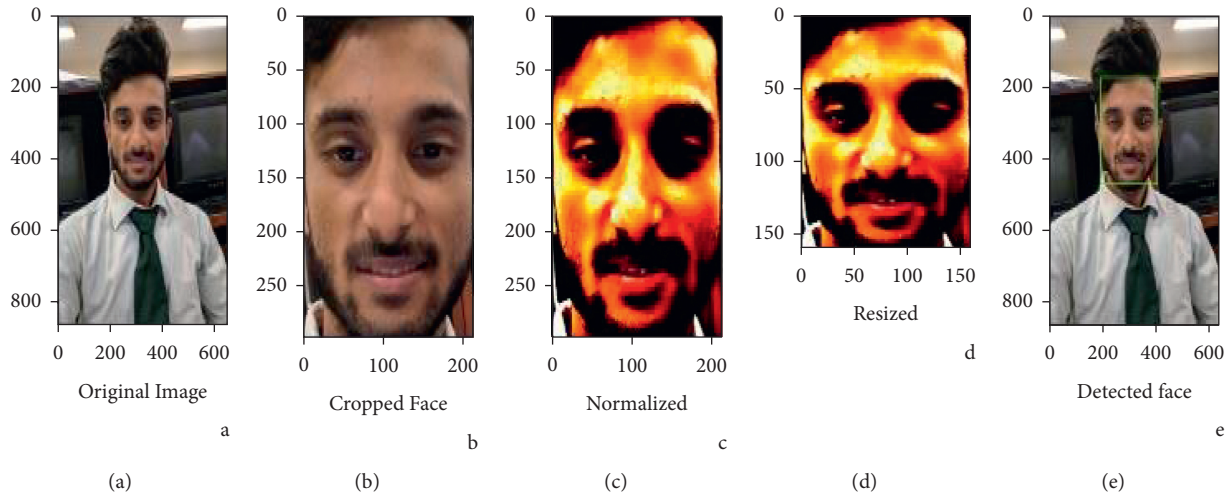


FIGURE 1: (a) Original face image. (b) MTCNN algorithm minimized the unwanted features and returns cropped face. (c) Normalized cropped face. (d) Resized image to 160×160 . (e) Face recognized with Bounded Box.

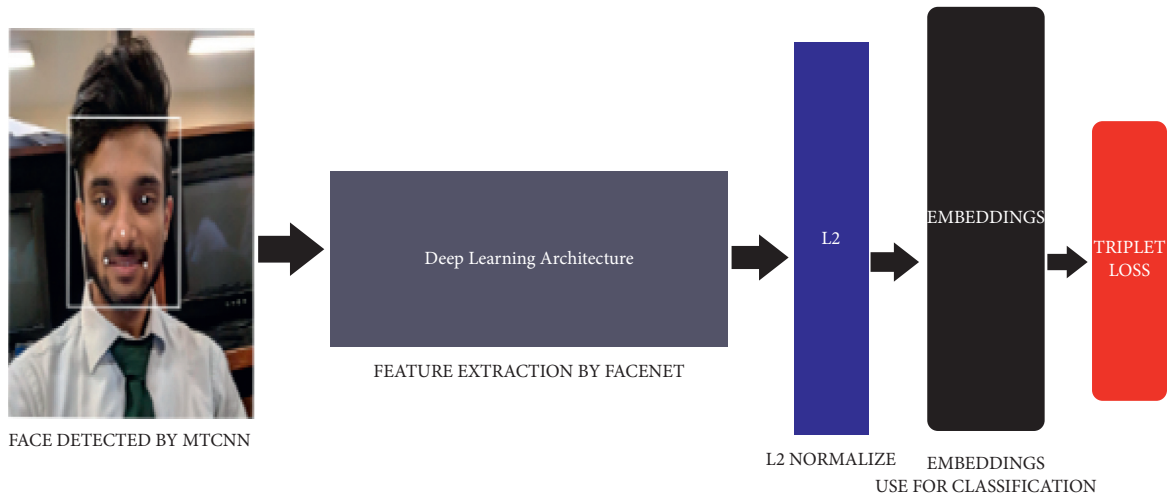


FIGURE 2: The FaceNet architecture receives the input image after the face has been detected by MTCNN, where the deep learning architecture is used to extract the best features followed by L2 normalization. The result of this architectural processing is facing embeddings that are forwarded to the triplet loss function during training.

separation can be done linearly or nonlinearly both. The proposed methodology compared the test face to other faces using the Support Vector Machine. The result is deemed correct if the distance between the image which we train, and the test of the same person is kept to a minimum. A facial resemblance is measured on the image which we have input and the image of the face formed by estimating a level 2 normalization in the characteristics of the specific points gathered from the network structure.

3. Proposed Management System

This process takes the recognized face which is delivered from face identification utilizing SVM. The name of the face owner will be marked present in our database with the current timestamp by the interfacing of Python with SQLITE3. The number of faces will be obtained through face detection which will be used at the cohort level or

individually for the management system. Figure 4 shows the approach of face recognition-based management system as explained in Section 3

The proposed deep net topology is simpler than prior models, and this allows that net to be extended into a deeper network in a straightforward way as shown in Figure 5.

3.1. Experimental Results. In the following, we initially tested the models and instructional data for the specific dissimilarity of the suggested models and instructional information using the Labeled Faces in Wild Dataset to verify performance. Next, we compared the performance of configuration with leading-edge methods on LFW (Labeled Faces in Wild Dataset). Our implementation is predicated in Python related to public libraries of NVIDIA CuDNN to boost the training. All experiments were carried out on NVIDIA GTX 1650 with 4 GB of onboard memory. This is often important

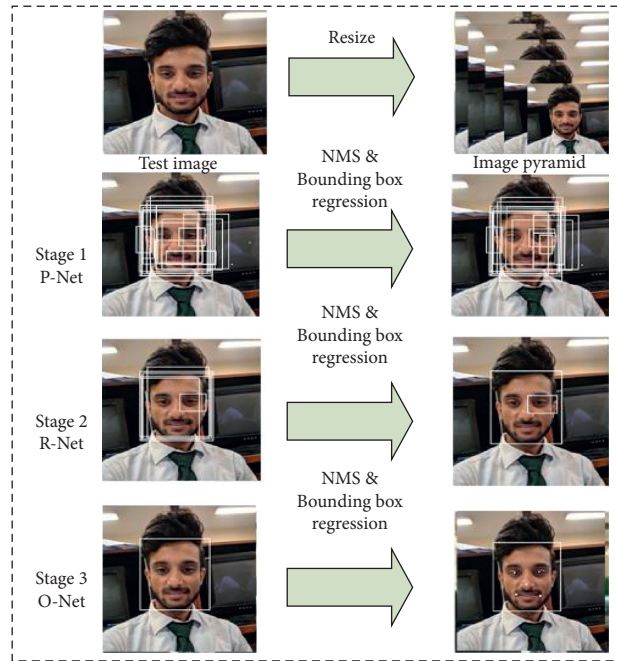


FIGURE 3: Face detection with facial landmarks.

due to the limited memory footprint and great complexity of very deep networks. For multiscale testing, the face is initially resized to 3 different sizes of 256, 384, and 512 pixels. Accordingly, the cropping method was repeated for each of them. The overall outcome size descriptor is the overall mean for these characteristic vectors. Faces are identified with the help of the methodology explained in “FaceNet: a unified embedding for face recognition and clustering.

Figure 6 represents the vector of 128 numbers which represent the most important features of each tested face. This vector is further converted into 128 distinct embeddings using L2 distance measures between the test faces which will be used to identify the test subject. This means that, for example, once Rajal’s image is taken and converted into the facial feature vector, if the vector has a small distance with his distinct embeddings (prestored), then it signifies that his face has been identified. However, if, for example, Hamza’s facial feature vector has a larger distance from the distinct embeddings based on the measures, then it means that his face has not been identified and another picture needs to be taken. The figure shows the difference in the embeddings between the 4 detected faces of our dataset which have been used as the input to our classifier model.

The identification accuracy from the distance of 1 m to 5 m from the camera on different scenarios is shown in Table 2.

For identifying, the detected faces are first converted into 128 distinct embeddings. The correlation of these embeddings is taken to represent the distance for recognition in which the Minimum Threshold of the distance is taken as 0 while the maximum is taken as 0.6 (60%) out of 1 (100%). It can be seen, in Figure 6, that for Sufyan, Waqas, and Hamza, the distance is within the “Maximum Threshold” and “Total Distance” bounds, thus indicating that their faces have been

detected with an adequate level of certainty. However, for Rajal, the certainty to which his face has been detected is low since it is below the “Maximum Threshold” despite being above the “Minimum Threshold.” This can be attributed to various aspects such as a blurry image or a major difference between the facial features of the taken and reference image. If the embedding is within threshold bounds, then it indicates that the image has been identified to a certain degree as shown in Figure 7.

Table 3 represents the distance refers to the dissimilarities of the detected image and image stored inside the database.

Table 4 provides the accuracy of detection from various angles. It represents face capturing angle which is one of the causes of face recognition. The results suggest that when pose variation is within 15°.

Figure 8 represents the similarities between the embeddings of the recognized face which were stored in the database and the input face which we acquired through real-time image acquisition.

The result shows that the pose variation at different angles has weight to improve the accuracy. Figure 9 shows that the pose of the face from -15 to $+15^\circ$ provides high accuracy since in that position the features of the face can be detected easily, and face recognition is better as compared to other poses.

In this system, a pretrained model (MTCNN) is used to find the candidate’s face in part of the image and interpret it into greater feature facial descriptors. This network works as cascading in this model as shown in Figure 10.

This model is then optimized to not only improve accuracy under realistic conditions but also reduce computation time through postprocessing, feature

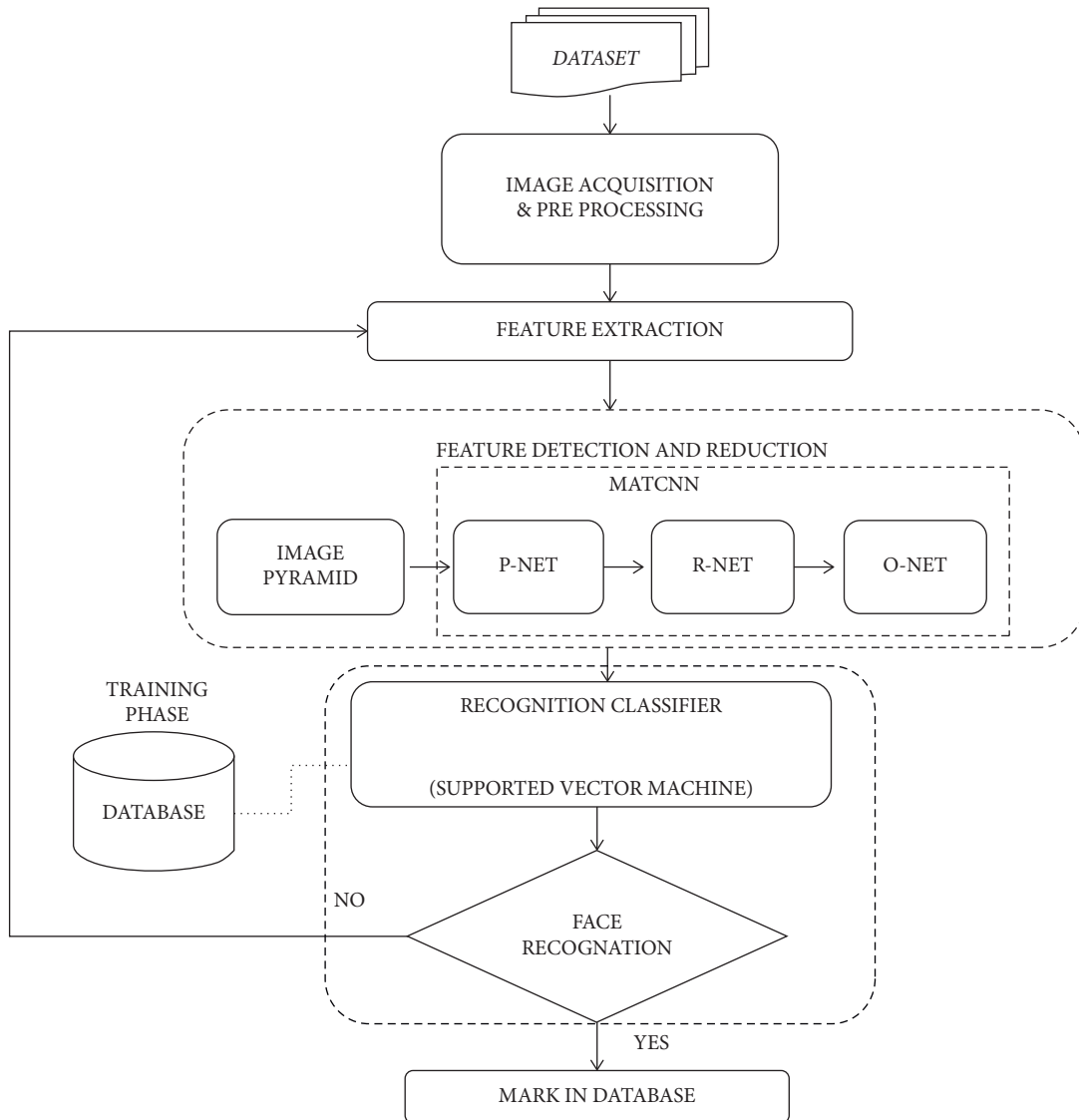


FIGURE 4: Flow of Smart Identity Management System by face detection.

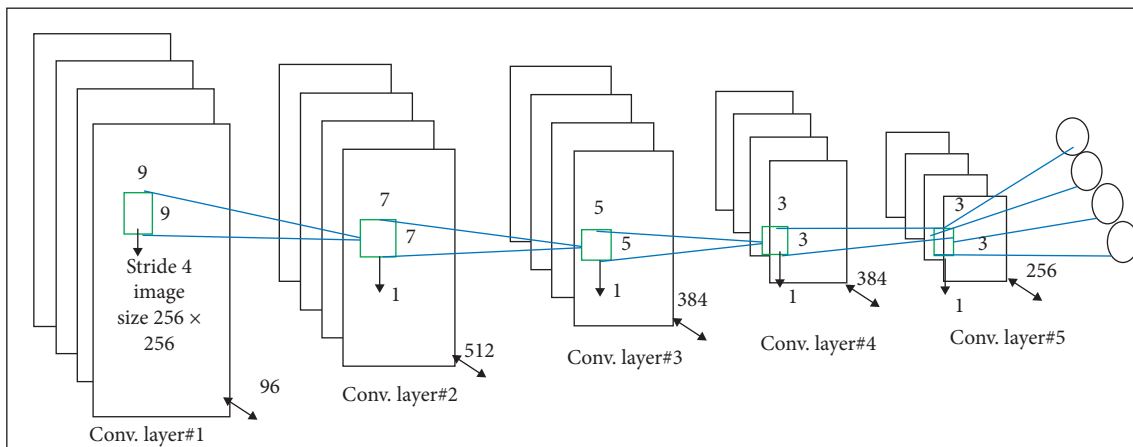


FIGURE 5: Architecture of deep net: high-order local and global features are learned in an image through multilayer architectures. Kernel size and quantity along with kernel combinations between layers determine the learning capacity of the networks.

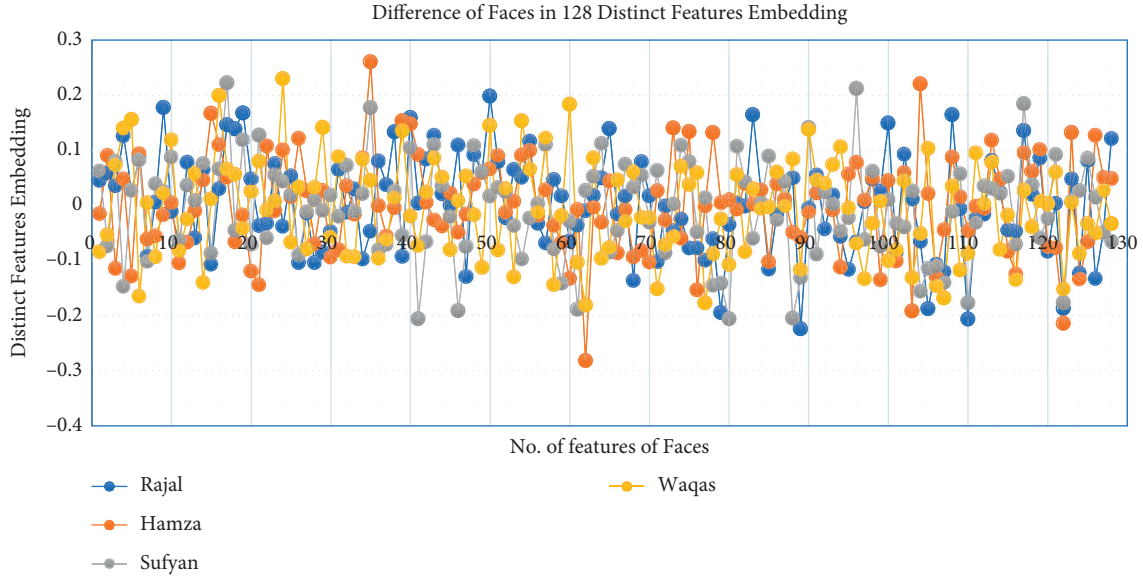


FIGURE 6: 128 distinct feature embeddings.

TABLE 2: Face Recognition Accuracy test from distance.

Scenario	Illuminance (lx, lumen/m ²)	Accuracy distance (1 m)	Accuracy distance (3 m)	Accuracy distance (5 m)	Avg. Accuracy (%)
Low light	100	99.630	97.88	92.0	96.47
Normal light	300	100	99.97	99.92	99.96
Bright	1000	99.00	99.00	94.00	97.33
Dark	>10	54.00	49.76	44	49.253

Bold values mean reference results at normal light scenario and are used as a comparison for other scenarios.

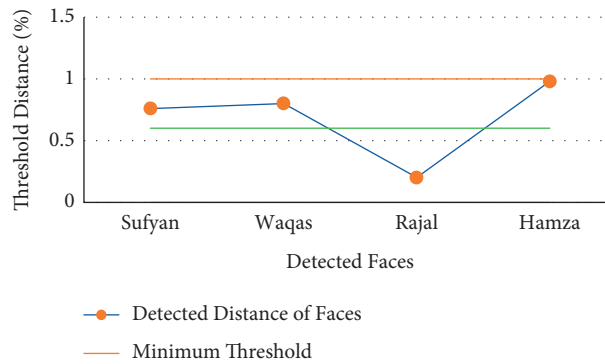


FIGURE 7: Recognition between threshold distances.

TABLE 3: Actual Distance from Total Distance for recognition.

Persons	Sufyan	Waqas	Rajal	Hamza
Actual Distance between encoding (m)	0.746564	0.80876	0.225128	0.941572
Total Distance (m)	1	1	1	1
Maximum Threshold	0.6	0.6	0.6	0.6
Minimum Threshold	0	0	0	0

extraction, and Multitasking Cascaded Convolution Neural Networks (MTCNN) algorithm. As a result, the computation time decreased to 0.073–0.40 s for facial recognition and identification. We also utilized a method

of having unified face image representation necessary for better recognition of face images. The system provides low-cost memory storage and has data logging features and low maintenance.

TABLE 4: Performance comparison from various angles.

Pose (degree)	-45	-30	-15	15	30	45
Accuracy (%)	84.5	95.2	97.1	98.8	95.8	89

Bold values mean good accuracy results at above mentioned angles.

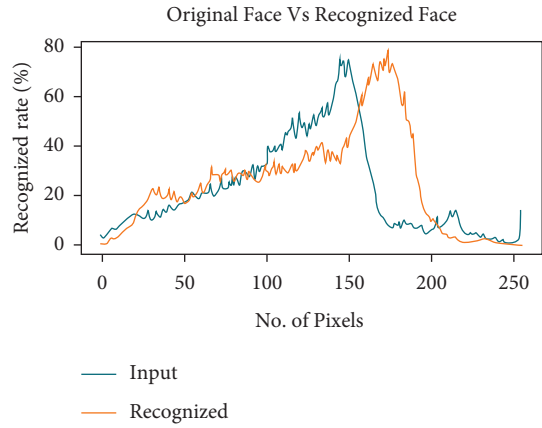


FIGURE 8: Detected face versus recognized face.

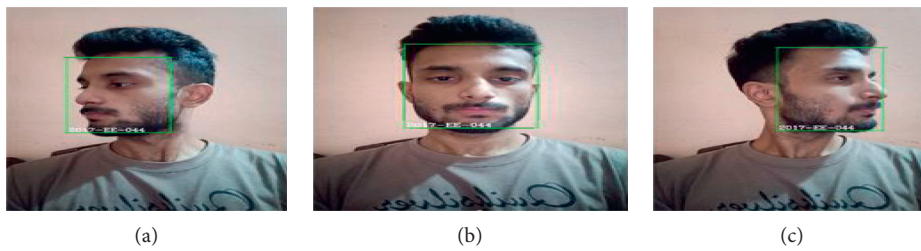


FIGURE 9: Recognition from varied angles.

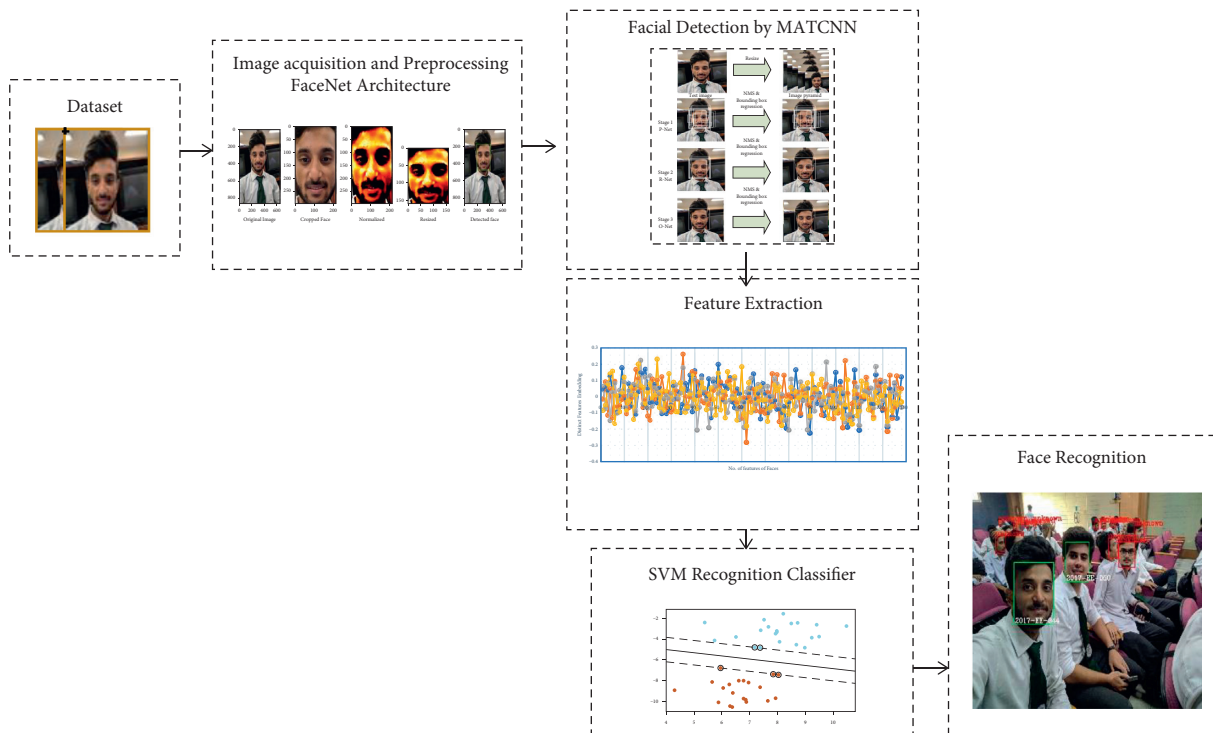


FIGURE 10: Structure of the deep network of Smart Identity Management System by face detection.

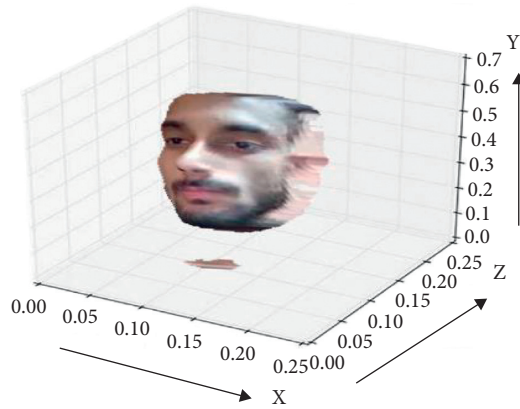


FIGURE 11: Face plotted in 3-dimension.

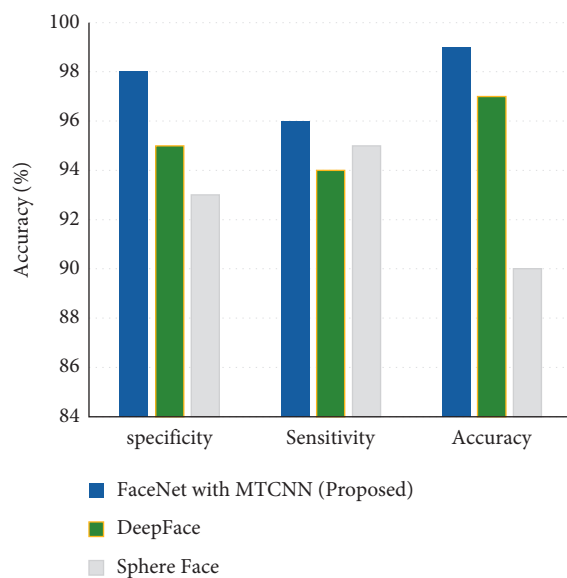


FIGURE 12: Performance test with other algorithms.

The result of our dataset performed really well in recognition from various angles. Adding additional images which were captured from the various angles makes our system recognition look similar to 3D recognition where a 3D model of a sample face is projected in Figure 11. It shows the actual projected image of the face in the x , y , and z axes.

Varying the algorithm on the system provides minimal variations to the performance of the system. Figure 12 shows the system's performance in terms of its accuracy, sensitivity, and specificity using the Deep Face, Sphere Face, and Proposed FaceNet with MTCNN hybrid Net. The proposed Net adapts to the performance of the system by having a standout result compared with the two other algorithms.

4. Conclusion

This proposed approach has the idea of implementing a smart system that is able to identify the face in real time while taking both luminosity and distance into account. It has been implemented on the features which are successfully able to get results with 97.1% to 98.8% accuracy when the

position of the face is in the -15° to $+15^\circ$ range while other positions provide average results. This issue can be improved if we train our database with 3D images taken by a 3D scanner or camera which will boost the performance of recognition and increase the range of which algorithm can identify faces accurately. This identification system is able to recognize a face accurately with 99% to 98% accuracy when the distance of face is 4-5 meters away from the camera under normal light conditions. Additionally, under low light conditions, the average accuracy achieved under low light conditions is 96.47%. The recognition range can be increased by using a high-quality camera that is capable of HD imaging. The approach was also compared with two other industry-standard architectures used for facial detection and it was observed that FaceNet with MTCNN had the highest performance in terms of specificity, sensitivity, and accuracy. In the proposed methodology, postprocessing of images with feature extraction and algorithms reduced the face detection computational time and improved the face identification accuracy. The system was able to accurately identify faces from distances up to 5 m and in both high to low light intensities of the local environment.

Data Availability

The authors used third-party data and do not have the rights to share it. The third-party data cannot be publicly shared. Researchers must request to gain access to the data in which case the authors will apply to gain access and share it with them.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] S. Chintalapati and M. V. Raghunadh, "Automated attendance management system based on face recognition algorithms," in *Proceedings of the 2013 IEEE International Conference on Computational Intelligence and Computing Research*, pp. 1–5, IEEE, Enathi, India, December 2013.
- [2] A. Ezzahout and R. O. H. Thami, "Conception and development of a video surveillance system for detecting, tracking and profile analysis of a person," in *Proceedings of the 2013 3rd International Symposium ISKO-Maghreb*, pp. 1–5, IEEE, Marrakech, Morocco, November 2013.
- [3] L. Zhang, J. X. Wang, and K. Zhang, "Design of embedded video monitoring system based on S3C2440," in *Proceedings of the 2013 Fourth International Conference on Digital Manufacturing and Automation*, pp. 461–465, IEEE, Shinan, China, June 2013.
- [4] A. Singh, D. Patil, M. Reddy, and S. N. Omkar, "Disguised Face Identification (DFI) with facial key points using spatial fusion convolutional network," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1648–1655, IEEE, Venice, Italy, October 2017.
- [5] Y. Yu, X. Duan, S. Wang, and B. Jiao, "The design and implementation of Bluetooth video surveillance devices," in *Proceedings of the 2010 3rd International Congress on Image*

- and Signal Processing*, pp. 495–498, IEEE, Yantai, China, October 2010.
- [6] E. Jose, M. Greeshma, M. T. Haridas, and M. H. Supriya, “Face recognition-based surveillance system using Face-Net and MTCNN on Jetson TX2,” in *Proceedings of the 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, pp. 608–613, IEEE, Coimbatore, India, March 2019.
- [7] R. G. Cutler, *Face Recognition Using Infrared Images and Eigen Faces*, University of Maryland, College Park, MD, USA, 1996.
- [8] I. Intan, “Combining of feature extraction for real-time facial authentication system,” in *Proceedings of the 2017 5th International Conference on Cyber and IT Service Management (CITSM)*, pp. 1–6, IEEE, Denpasar, Indonesia, August 2017.
- [9] F. J. Prokoski, R. B. Riedel, and J. S. Coffin, “Identification of individuals by means of facial thermography,” in *Proceedings of the 1992 International Carnahan Conference on Security Technology: Crime Countermeasures*, pp. 120–125, IEEE, Atlanta, GA, USA, October 1992.
- [10] T. Elguebaly and N. Bouguila, “A Bayesian method for infrared face recognition,” in *Machine Vision beyond Visible Spectrum* Springer, Berlin, Heidelberg, Germany, 2011.
- [11] Z. Lin, Z. Wenrui, S. Li, and Z. Fang, “Infrared face recognition based on compressive sensing and PCA,” in *Proceedings of the 2011 IEEE International Conference on Computer Science and Automation Engineering*, pp. 51–54, IEEE, Shanghai, China, June 2011.
- [12] D. Polsgrove and C. Woods, “Biometric verification of visible and MWIR images suitable for optical correlators,” in *Frontiers in Optics* Optical Society of America, Washington, DC, USA, 2004.
- [13] R. S. Ghiass, O. Arandjelović, A. Bendada, and X. Maldague, “Infrared face recognition: a comprehensive review of methodologies and databases,” *Pattern Recognition*, vol. 47, no. 9, pp. 2807–2824, 2014.
- [14] S. Moon, S. G. Kong, J. H. Yoo, and K. Chung, “Face recognition with multiscale data fusion of visible and thermal images,” in *Proceedings of the 2006 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, pp. 24–27, IEEE, Alexandria, VA, USA, October 2006.
- [15] X. Chen, P. J. Flynn, and K. W. Bowyer, “IR and visible light face recognition,” *Computer Vision and Image Understanding*, vol. 99, no. 3, pp. 332–358, 2005.
- [16] X. Chen, Z. Jing, and G. Xiao, “Fuzzy fusion for face recognition,” *International Conference on Fuzzy Systems and Knowledge Discovery*, Springer, Berlin, Heidelberg, Germany.
- [17] C. Zhu, Y. Zheng, K. Luu, and M. Savvides, “CMS-RCNN: contextual multi-scale region-based CNN for unconstrained face detection,” in *Deep Learning for Biometrics* Springer, Cham, Switzerland, 2017.
- [18] E. Sun, “Small-scale image recognition based on cascaded convolutional neural network,” in *Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 2737–2741, IEEE, Chongqing, China, 12 March 2021.
- [19] X. Li, Z. Du, Y. Huang, and Z. Tan, “A deep translation (GAN) based change detection network for optical and SAR remote sensing images,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 179, pp. 14–34, 2021.
- [20] A. M. Faruk, H. A. Faraby, M. Azad, M. Fedous, and M. Morol, “Image to Bengali caption generation using deep CNN and bidirectional gated recurrent unit,” 2020, <https://arxiv.org/abs/2012.12139>.
- [21] R. Alimuin, E. Dadios, J. Dayao, and S. Arenas, “Deep hypersphere embedding for real-time face recognition,” *Telkomnika*, vol. 18, no. 3, pp. 1671–1677, 2020.
- [22] R. Jin, H. Li, J. Pan, W. Ma, and J. Lin, “Face recognition based on MTCNN and Facenet,” 2021, https://jasonyanglu.github.io/files/lecture_notes/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0_2020/Project/Face%20Recognition%20Based%20on%20MTCNN%20and%20FaceNet.pdf.
- [23] J. Xiao, J. Wang, S. Cao, and Y. Li, “Research on the improvement of MTCNN ALGORITHM: face recognition with mask,” in *Proceedings of the 2021 2nd International Conference on Artificial Intelligence and Information Systems*, pp. 1–7, ACM, Chongqing China, May 2021.
- [24] J. Heo, S. G. Kong, B. R. Abidi, and M. A. Abidi, “Fusion of visual and thermal signatures with eyeglass removal for robust face recognition,” in *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop*, p. 122, IEEE, Washington, DC, USA, June 2004.
- [25] S. G. Kong, J. Heo, F. Boughorbel et al., “Multiscale fusion of visible and thermal IR images for illumination-invariant face recognition,” *International Journal of Computer Vision*, vol. 71, no. 2, pp. 215–233, 2007.
- [26] M. Ma and J. Wang, “Multi-view face detection and landmark localization based on MTCNN,” in *Proceedings of the 2018 Chinese Automation Congress (CAC)*, pp. 4200–4205, IEEE, Xi’an, China, November 2018.
- [27] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.
- [28] W. Hizem, L. Allano, A. Mellakh, and B. Dorizzi, “Face recognition from synchronised visible and near-infrared images,” *IET Signal Processing*, vol. 3, no. 4, pp. 282–288, 2009.

Research Article

Linkable Ring Signature Scheme Using Biometric Cryptosystem and NIZK and Its Application

Xuechun Mao , Lin You , Chengtang Cao , Gengran Hu , and Liqin Hu 

College of Cybersecurity, Hangzhou Dianzi University, Hangzhou, China

Correspondence should be addressed to Lin You; mryoulin@gmail.com

Received 16 April 2021; Revised 16 September 2021; Accepted 27 October 2021; Published 15 December 2021

Academic Editor: David W. Chadwick

Copyright © 2021 Xuechun Mao 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.

Biometric encryption, especially based on fingerprint, plays an important role in privacy protection and identity authentication. In this paper, we construct a privacy-preserving linkable ring signature scheme. In our scheme, we utilize a fuzzy symmetric encryption scheme called symmetric keyring encryption (SKE) to hide the secret key and use non-interactive zero-knowledge (NIZK) protocol to ensure that we do not leak any information about the message. Unlike the blind signature, we use NIZK protocol to cancel the interaction between the signer (the prover) and the verifier. The security proof shows that our scheme is secure under the random oracle model. Finally, we implement it on a personal computer and analyze the performance of the constructed scheme in practical terms. Based on the constructed scheme and demo, we give an anonymous cryptocurrency transaction model as well as mobile demonstration.

1. Introduction

With the advantages of decentralized control and anonymous payment, cryptocurrency is gradually replacing the traditional payment mode. However, the anonymity provided by bitcoin has been questioned in the sense that it offers pseudonymity instead of real anonymity. The research work [1] has shown that attackers can improperly obtain the actual identity of a bitcoin's owner or even other users through proxy addresses. In order to improve anonymity, researchers have proposed various privacy protection schemes, such as Dash based on the mixed coins protocol, Monero based on the CryptoNote protocol, and Zerocoin [2] based on the Zero-Knowledge Succinct Non-interactive Arguments of Knowledge (zk-SNARK) protocol [3], etc.

In 2015, Noether [4] improved Monero's original CryptoNote protocol by using a variant of linkable ring signature, which was called Ring Confidential Transactions (Ring CT). In the Ring CT protocol, Noether improved the "one-time ring signature" to linkable ring signature as the core cryptoprimitive to provide anonymity, which could not

only meet the actual transaction needs, but also prevent the occurrence of double spending in transactions. Meanwhile, Monero also used stealth address to hide the recipient's identity. Sasson et al. [2] proposed Zerocash, which used zk-SNARK protocol to construct the anonymous electronic cash system to protect the privacy of users and transaction amounts.

Along with the development of cryptocurrency applications, privacy protection has gradually become an important issue. How to authenticate users while ensuring their anonymity has always been an important challenge in the information age. Biometric encryption technology which combines the cryptographic schemes with biometrics is an important branch of biometric protection technology. It is designed to protect secrets by binding/retrieving secrets with biometrics rather than using passwords or tokens in conventional cryptographic systems. Compared with passwords or tokens, biometrics such as fingerprints are more convenient, stable, and unforgettable. Nowadays, many biometric encryption algorithms have been proposed [5, 6]. In a word, combining biometric encryption technology and

NIZK protocol with ring signatures will provide a great potential advantage for the protection of users' privacy.

1.1. Related Work

Linkable Ring Signature Schemes. In 2004, linkability property was first introduced in a ring signature scheme by Liu et al. [7]. Later, Franklin and Zhang [8] proposed a general framework for linkable ring signatures. Nowadays, there were many variants of linkable ring signatures based on different features. Deng et al. [9] presented a new identity-based linkable ring signature scheme which avoided certificate management. Sun et al. [10] formalized the syntax of Ring CT protocol and then put forward a new efficient Ring CT protocol (Ring CT 2.0) which could save significant space.

Non-Interactive Zero-Knowledge (NIZK) Protocol. In 1988, Blum et al. [11] firstly studied the NIZK proof system and presented the common reference string model which is generally applied to Zerocash. This construction is a NIZK range proof system based on a number theoretic assumption related to factoring. Using the Fiat-Shamir heuristic, Groth [12] suggested a NIZK argument for correctness of an approval vote. In order to optimize the size of NIZK proofs, Gentry et al. [13] constructed a fully homomorphic hybrid encryption scheme to minimize the communication cost. In 2019, Tsai et al. [14] proposed a new non-interactive ZKRP scheme to maintain high flexible range form.

Biometric Encryption Schemes. In 1994, combining with fingerprint recognition, Tomoko firstly proposed the concept of biometric encryption and applied it for patents. Since then, various biometric cryptographic algorithms have been proposed. Juels and Sudan [15] put forward the concept of fuzzy vault whose security is based on the hardness of the polynomial reconstruction. However, Osadchy and Dunkelmann [16] found that many of the existing schemes do not consider the privacy and security aspects of the feature extraction and binarization processes which have a huge risk for user privacy. Therefore, Lai et al. [5] proposed a novel biometric cryptosystem for vectorial biometrics called symmetric keyring encryption (SKE) by using an index of maximum hashed vectors, simple filtering mechanism, and Shamir's secret-sharing scheme. They also formalized and analyzed the threat model for SKE, which involved four major security attacks.

1.2. Contributions. In this work, we firstly use a simplistic biometric secret-binding scheme called SKE to encrypt the secret key which can protect user's secret key and authenticating user's identity at the same time. Second, we utilize a NIZK protocol to provide anonymity for the message. Unlike the blind signatures [17, 18] which have similar property, no interaction is required during the signing and validation process of our scheme. The security analysis shows that our proposed scheme is provably secure under the random oracle model. Third, we analyze the

performance of the proposed scheme and also implement it based on the fingerprint model. The encouraging results indicate that our scheme is practicable. Finally, we propose an anonymous cryptocurrency transaction model with a corresponding mobile demo.

1.3. Structure of the Paper. The rest of this paper is organized as follows. In Section 2, some notations are introduced, and SKE and NIZK protocols are described. System framework and security model are presented in Section 3. We describe the signature scheme in Section 3. And its security analysis is provided in Section 5. In Section 6, some experimental results are given. The anonymous cryptocurrency transaction model is given in Section 7. Finally, the last section gives the conclusion.

2. Preliminary

First, we give some notations in Table 1 which are used in the rest of this paper.

2.1. SKE Model. The SKE model is a novel simplistic biometric secret-binding scheme for vectorial biometrics which is based on the notion of symmetric key cryptosystems [5]. First, the SKE model uses IoM hashing [19] which can generate abundant IoM hashed entries as genuine entries without being restricted by the original biometric vector size. We use $x \in \mathbb{Z}_q^d$ as an enrolled biometric vector and N as a random projection matrix. Given x, m, N , the IoM hashing operations as follows: $\Omega^{\text{IoM}}(x, m, N) \rightarrow \phi_x \in \mathbb{Z}_q^m$.

During enrollment, there is a user with an enrolled biometric vector x , parameter m , two different nonces N, \tilde{N} , and a finite field polynomial $f(\cdot) \in \mathcal{F}_q^m$ of order to generate ϕ_x and $\tilde{\phi}_x$. Then, we perform polynomial projection to generate $f(\phi_x)$ and yield a public secure sketch $ss = (\tilde{\phi}_{x(1)} \oplus f(\phi_{x(1)}), \dots, \tilde{\phi}_{x(m)} \oplus f(\phi_{x(m)}))$. Finally, we generate authentication tag $\text{TAG} = (\text{tag}_1, \dots, \text{tag}_m)$, where $\text{tag}_i = H(\phi_{x(i)} \| ss_i \| \tilde{\phi}_{x(i)})$ is the one-way hashed output. We store $(N, \tilde{N}, \text{TAG}, ss)$ as the public helper data.

During secret retrieval, given the query biometric vector x' and the public helper data above, we generate $\phi_{x'}$ and $\tilde{\phi}_{x'}$ as well as tag'_i and TAG' following the steps above.

When $\text{tag}'_i = \text{tag}_i$, we can get a genuine pair $(\phi_{x(i)}, f(\phi_{x(i)})) \in G$, where G is a genuine set. If we have a sufficient number of revealed genuine pairs with $|U| = t \geq k$, the secret key y can be retrieved via polynomial interpolation using the unlocking set $U = \{(\phi_{x(i)}, f(\phi_{x(i)}))\}_{i=1}^t$. A high-level overview of SKE is shown in Figure 1.

2.2. NIZK Protocol. NIZK protocols can be used to demonstrate the truth of a statement without revealing anything else. We briefly state the NIZK protocol [12] which we will use below. First, we set the system parameters $g, h \in \mathbb{G}$. We also denote $H_1(\cdot), H_2(\cdot)$ to be secure hash functions and individually output $\mu, e \in \mathbb{Z}_q$. Randomizer R and message are prover's input. Then, we simply discuss this NIZK as follows.

Definition 2 (NIZK Argument of Knowledge [20]). A NIZK argument of knowledge for a relation R is a NIZK argument for R with the following additional extractability property.

(1) *Extraction.* For any PPT adversary \mathcal{A} , random string $r \leftarrow \{0, 1\}^*$, there exists a PPT algorithm Ext outputting w' such that the following probability is negligible in κ :

$$\Pr(\text{Verify}(\Sigma, \tilde{s}, \tilde{\pi}) = 1 \wedge R(\tilde{s}, w') = 0). \quad (3)$$

2.3. (t, u) -Threshold Secret-Sharing Scheme. In this section, we describe a (t, u) -threshold scheme [21] which enables making u shares (distribution) and recovering the secret from t or more shares (recovery) using just XOR operations, for arbitrary threshold t and the number of participants u . We will only use the distribution algorithm which is described in Algorithm 1. In this algorithm, the secret $y \in \{0, 1\}^{d(u_p-1)}$ needs to be divided equally into $u_p - 1$ blocks $y_1, \dots, y_{u_p-1} \in \{0, 1\}^d$, where u_p is a prime number, and $d > 0$ denotes the bit-size of every divided piece of the secret. Also, it uses u shares, y_1, \dots, y_u , of a (t, u_p) -threshold scheme to construct a (t, u) -threshold scheme if the desired number of participants u is a composite number (in our scheme, we set $u = u_p$).

These XORed terms are circulated in a specific pattern with t dimensions and do not overlap with each other because the properties of prime numbers are used. By an implementation on a PC, they showed that the proposed scheme is able to make u shares from the secret and recover the secret from t shares more quickly than Shamir's scheme [22] if u is not extremely large.

2.4. UTXO Ledger Model. Bitcoin, the most valuable and popular cryptocurrency, uses a graph-based ledger model built on the concept of UTXOs (unspent transaction outputs). In the UTXO ledger model, individual transactions consist of a list of inputs and a list of outputs. Each of the transactions can merge the bitcoins in the previous multiple accounts and transfer them to another one or more accounts. Figure 2 shows how UTXO model works, where Tx1 contains one input and two outputs, and Tx2 contains three inputs and two outputs.

2.5. Linkable Ring Signature. The biometric cryptosystem can be found in [5] and the NIZK scheme can be found in [12]. Our definitions are in the spirit of [4, 5, 12].

Definition 3. A linkable ring signature scheme based on SKE and NIZK consists of five algorithms:

- (1) **Setup:** on input of the user's biometric vector x and a finite field polynomial order $k - 1$, output public helper data PP .
- (2) **KeyGen:** on input of the user's biometric vector x' and public helper data PP , output the secret key-vectors y and its corresponding public key-vectors PK .

- (3) **Sign:** on input of a message, the parameters g, h, R, G , and the set $L = \{PK_i^j\}_{i \in [n]}^{j \in [u]}$ where (y_j, PK_π^j) is a valid key pair output by KeyGen and $PK_\pi^j \in L$, output a signature Ω .
- (4) **Verify:** on input of the purported signatures Ω , anyone can verify Ω and output a bit $b \in \{0, 1\}$.
- (5) **Link:** on input of two messages M_1, M_2 as well as two signatures Ω_1 and Ω_2 , output a bit $b \in \{0, 1\}$.

2.6. Complexity Assumptions and Lemma

Definition 4 (Discrete Logarithm (DL) Assumption). Given a generator g of G^* , where $G^* = \mathbb{G}$ or \mathbb{G}_T , and $a \in \mathbb{Z}_q^*$, for every adversary \mathcal{A} , $\Pr[\mathcal{A}(g, ag) = a] = \varepsilon(\kappa)$.

Definition 5 (Decision Diffie-Hellman (DDH) Assumption). Distinguish the distributions (ag, bg, abg) and (ag, bg, cg) with $a, b, c \leftarrow \mathbb{Z}_q$ and $g \in \mathbb{G}$. The DDH assumption is the intractability of the problem for any PPT distinguisher.

Lemma 1 (From Liu et al. [7]). Let \mathcal{A} be an attacker and \mathcal{C} be a challenger; \mathcal{C} invokes \mathcal{A} to obtain a transcript \mathcal{T} ; if \mathcal{T} is successful, then \mathcal{C} rewinds \mathcal{T} to a header \mathfrak{H} and resimulates \mathcal{A} to obtain transcript \mathcal{T}' . If $\Pr(\mathcal{T} \text{ succeeds}) = \varepsilon$, then $\Pr(\mathcal{T}' \text{ succeeds}) = \varepsilon$.

3. Security Model

In consideration of the security, our scheme should satisfy four fundamental properties: unforgeability, anonymity, linkability, and zero-knowledge which are very similar to the definitions given by [4, 7].

Before giving the definition, we give the definitions of the following queries at first. They will be carried out between a challenger \mathcal{C} and an adversary \mathcal{A} , which together simulate the ability of the adversary.

- (1) **Hash functions query:** \mathcal{A} may request the values of the hash functions for any input.
- (2) **Key query:** \mathcal{A} requests the key of a user; \mathcal{C} responds with the secret key.
- (3) **Signature query:** \mathcal{A} submits a tuple (M, x, L) ; \mathcal{C} outputs a signature.

3.1. Unforgeability. We give the adversary model about unforgeability, which follows a similar structure as [4, 7].

For any PPT adversary \mathcal{A} , the advantage that \mathcal{A} wins the following game can be ignored; then our scheme is said to be unforgeable.

Game I: an adversary \mathcal{A} plays a game with a challenger \mathcal{C} as follows.

Initialization: running the Setup algorithm, \mathcal{C} obtains the public helper data PP and then gives it to \mathcal{A} .

Query: \mathcal{A} performs a polynomially bounded number of queries.

```

Input:  $y \in \{0, 1\}^{d(u_p-1)}$ .
Output:  $y = (y_1, \dots, y_u)$ 
(1)  $s = 0^d, s_1 \parallel \dots \parallel s_{u_p-1} = s$ .
(2) for  $1 \leq i \leq t-1$  do
(3) for  $1 \leq j \leq u_p$  do
(4) Choose  $r_j^i = \text{GEN}(\{0, 1\}^d)$ ;
(5) end
(6) end
(7) for  $1 \leq i \leq u$  do
(8) for  $1 \leq j \leq u_p-1$  do
(9) Choose  $y_{(i,j)} = (\oplus_{h=1}^{t-1} r_{hi+j}^h) \oplus s_{j-i}$ ;
(10) end
(11)  $y_i = y_{(i,1)} \parallel \dots \parallel y_{(i,u_p-1)}$ 
(12) end
(13) return  $y = (y_1, \dots, y_u)$ .

```

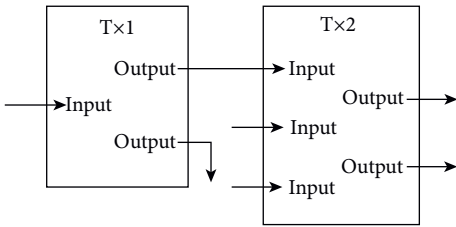
ALGORITHM 1: DisThreshold (y, t, u).

FIGURE 2: UTXO ledger model.

Forge: \mathcal{A} submits a new tuple $(\Omega^*, x^*, M^*, L^*)$. \mathcal{A} will win if the following conditions hold:

- (1) Ω^* is a legal signature.
- (2) \mathcal{A} did not query the key of anyone in L^* .
- (3) \mathcal{A} did not query the tuple (x^*, M^*) .

The advantage of the unforgeability is denoted by

$$\text{Advantage}_{\mathcal{A}}^{\text{Forge}} = \Pr[\mathcal{A} \text{ succeeds}]. \quad (4)$$

If for any PPT algorithm \mathcal{A} , the advantage of $\text{Advantage}_{\mathcal{A}}^{\text{Forge}}$ is negligible, we say the scheme is unforgeable.

3.2. Anonymity. Our scheme is said to be signer anonymity if for any PPT adversary \mathcal{A} , $\text{Advantage}_{\mathcal{A}}^{\text{Forge}}$ is negligible.

Game II: an adversary \mathcal{A} plays a game with a challenger \mathcal{C} as follows.

Initialization: it is the same as that in Game I.

Query: \mathcal{A} performs a polynomially bounded number of queries.

Challenge: \mathcal{A} outputs a new tuple $(M, x_{i,0}, x_{i,1}, L)$. \mathcal{C} flips a coin $b \in \{0, 1\}$ and then returns \mathcal{A} with the signature $\Omega(M, x_{i,b}, L)$.

Guess: \mathcal{A} outputs a bit b' . If $b = b'$, \mathcal{A} is considered to succeed with the probability of $\text{Success}_{\mathcal{A}}^{\text{Anon}}$.

The anonymity advantage of our scheme is denoted by $\text{Advantage}_{\mathcal{A}}^{\text{Anon}} = |\text{Success}_{\mathcal{A}}^{\text{Anon}} - 1/2|$.

3.3. Linkability. If for any PPT adversary \mathcal{A} , the advantage of \mathcal{A} is ignorable in the following game, our scheme is said to be linkable.

Game III: an adversary \mathcal{A} plays a game with a challenger \mathcal{C} as follows.

Initialization, Query: it is the same as that in Game I.

Unlink: \mathcal{A} outputs two valid signatures (Ω_1, M_1, L_1) and (Ω_2, M_2, L_2) with respect to secret keys y_1 and y_2 , respectively. \mathcal{A} wins if the following conditions hold:

- (1) $\text{Verify}(\Omega_1, M_1, L_1) = \text{Verify}(\Omega_2, M_2, L_2) = 1$.
- (2) $y_1 \cap y_2 \neq \emptyset$.
- (3) $\text{Link}(\Omega_1, \Omega_2) = \text{unlink}$.

The advantage of the linkability is denoted by $\text{Advantage}_{\mathcal{A}}^{\text{Link}} = \Pr[\mathcal{A} \text{ wins}]$.

3.4. NIZK Argument. In ROM, if our scheme is proved to be completeness, computational soundness, and zero-knowledge, our protocol is a NIZK argument where the plaintext space is \mathbb{Z}_q .

4. Signature Scheme

4.1. Our Construction. The detailed steps of our scheme are given as follows.

Setup. (x)

On input of $x \in \mathbb{Z}_q^d$, two random nonces $N, \hat{N} \in \mathbb{Z}_q^{mq \times d}$, parameter $m \in \mathbb{Z}_q$, a finite field polynomial $f(\cdot)$ which encodes the secret key y , and a one-way hash function $H: \{0, 1\}^* \rightarrow \{0, 1\}^l$, the Setup algorithm does as follows:

- (1) Run IoM (x) to generate vectors $\varphi_x \leftarrow \Omega^{\text{IoM}}(x, m, N)$ and $\hat{\varphi}_x \leftarrow \Omega^{\text{IoM}}(x, m, \hat{N})$, where $\varphi_x = (\varphi_{x(1)}, \dots, \varphi_{x(m)})$, $\hat{\varphi}_x = (\hat{\varphi}_{x(1)}, \dots, \hat{\varphi}_{x(m)})$.
- (2) Compute $f(\varphi_x) = (f(\varphi_{x(1)}), \dots, f(\varphi_{x(m)}))$.
- (3) For $i \in [m]$, compute $ss_i = \hat{\varphi}_{x(i)} \oplus f(\varphi_{x(i)})$, such that the secure sketch $ss = (s_1, \dots, s_m)$.
- (4) For $i \in [m]$, compute $\text{tag}_i = H(\varphi_{x(i)} \parallel ss_i \parallel \hat{\varphi}_{x(i)})$, such that $\text{TAG} = (\text{tag}_1, \dots, \text{tag}_m)$.

(5) Output the public helper data $PP = \{N, \widehat{N}, ss, TAG\}$.

A formal description of this algorithm is shown Algorithm 2.

KeyGen. (x')

On input $x' \in \mathbb{Z}_q^d$, $u \in \mathbb{Z}_q$, the public helper data $PP = \{N, \widehat{N}, ss, TAG\}$ with parameter m , generator $G \in \mathbb{G}$, and the set $U = \emptyset$. The KeyGen algorithm does as follows:

- (1) Run IoM(x') to generate vectors $\varphi_{x'} \leftarrow \Omega^{\text{IoM}}(x', m, N)$ and $\widehat{\varphi}_{x'} \leftarrow \Omega^{\text{IoM}}(x', m, \widehat{N}) \in \mathcal{F}_q^m$.
- (2) For $i \in [m]$, compute $\text{tag}_i' = H(\varphi_{x'(i)} \| ss_i \| \widehat{\varphi}_{x'(i)})$, such that $TAG' = (\text{tag}_1', \dots, \text{tag}_m')$.
- (3) For $i \in [m]$, compute $f(\varphi_{x(i)})' = ss_i \oplus \widehat{\varphi}_{x'(i)}$ while $\text{tag}_i' = \text{tag}_i$, and then add $(\varphi_{x(i)}, f(\varphi_{x(i)}))$ to an unlocking set U .
- (4) If $|U| \geq k$, perform polynomial reconstruction with U and output the secret key $y = f(0) \in \mathbb{Z}_q^*$; else repeat step 1.
- (5) Run DisThreshold($y, u-1, u$) to obtain y , output $y = (y_1, \dots, y_u)$.
- (6) For $j \in [u]$, compute $PK_j = y_j G \in \mathbb{G}$. The $PK = (PK_1, \dots, PK_u)$ and y are the public key-vectors and the secret key-vectors, respectively.

A formal description of this algorithm is shown as Algorithm 3.

Sign.

On input the message $\in \{0, 1\}^l$, generators $g, h, G \in \mathbb{G}$, $R \in \mathbb{Z}_q^*$, and public key-matrix $L = \{PK_1, \dots, PK_n\}$ ($PK_i = (PK_i^1, \dots, PK_i^u)$) containing n public key-vectors of length u , user's secret key y corresponding to PK_π . We do as follows:

- (1) Let $H_1(\text{message}) = \mu$, choose $R_\mu, R_R \leftarrow \mathbb{Z}_q^*$, compute $C = \mu g + R h$; $C_R = R_\mu g + R_R h$; $e = H_2(C, C_R)$.
- (2) Compute $M = e\mu + R_\mu$, $R' = eR + R_R$.
- (3) For $j \in [u]$, $i = 1, \dots, \widehat{\pi}, \dots, n$, choose $s_i^j \leftarrow \mathbb{Z}_q$; when $i = \pi$, choose $a_\pi^j \leftarrow \mathbb{Z}_q$.
- (4) For $j \in [u]$, compute the key image $I_j = y_j H_p(PK_\pi^j) \in \mathbb{G}$.
- (5) When $i = \pi$, for $j \in [u]$, compute $L_\pi^j = a_\pi^j G$; $R_\pi^j = a_\pi^j H_p(PK_\pi^j)$; $c_{\pi+1} = H_3(M, L_\pi^1, R_\pi^1, \dots, L_\pi^u, R_\pi^u)$.
- (6) When $i = \pi + 1, \dots, n, 1, \dots, \pi - 1$, for $j \in [u]$, compute $L_i^j = s_i^j G + c_i PK_i^j$; $R_i^j = s_i^j H_p(PK_i^j) + c_i I_j$; $c_{i+1} = H_3(M, L_i^1, R_i^1, \dots, L_i^u, R_i^u)$.
- (7) When $i = \pi$, for $j \in [u]$, compute $s_\pi^j = a_j - c_\pi y_j \pmod q$.
- (8) Output $\Omega(M) = (I_j, c_1, s_i^j, C_R, M, R')_{i \in [n], j \in [u]}$.

A formal description of this algorithm is shown as Algorithm 4.

Verify

- (1) The verifier computes $e = H_2(C, C_R)$. Then, checking whether $eC + C_R = Mg + R'h$. If the congruence holds, return "1", otherwise "0".

- (2) For $i \in [n]$, $j \in [u]$, the verifier regenerates all the L_i^j, R_i^j, c_{i+1} and verifies whether $c_{n+1} = c_1$. If the congruence holds, return "1", otherwise "0".

A formal description of this algorithm is shown Algorithm 5.

Link.

For a fixed set of public key-vectors, given two messages M, M' , two valid signatures $\Omega(M)$ and $\Omega'(M')$, the verifier outputs *link* if the key image $I_j = I'_j$ ($j \in [u]$); otherwise the verifier outputs *unlink*.

5. Security Analysis

In this section, the security proofs of our scheme are given, which follow similarly to the security proofs of [4, 7].

Theorem 1. *In ROM, the scheme is unforgeable if DL problem is hard.*

Proof. (Similar proof to Theorem 6 of [4] and Theorem 1 of [7]) We follow the notation introduced above. Suppose that an adversary \mathcal{A} can forge signature with nonnegligible probability; then we certainly can construct a PPT simulator \mathcal{C} which can extract a solution to the DL problem. Given a random instance of the DL problem with security level κ , \mathcal{C} is asked to solve the DL problem in polynomial time.

First of all, \mathcal{C} maintains 4 lists $\ell_1, \ell_2, \ell_3, \ell_4, \ell_5$ in its local storage to store the outputs of H -oracle, H_1 -oracle, H_3 -oracle, key query, and sign query, initially setting to be empty. The interaction between \mathcal{A} and \mathcal{C} does as follows:

Initialization: Running the *Setup* algorithm; \mathcal{C} gives \mathcal{A} parameters.

Query: \mathcal{A} is allowed to make the following queries (supposed that \mathcal{A} will not initiate repeated queries).

- (1) H query: \mathcal{C} maintains list ℓ_1 of tuple (x_i, TAG_i) . For a request of x_i , if x_i contains list ℓ_1 , \mathcal{C} returns the corresponding tuple to \mathcal{A} ; otherwise, \mathcal{C} randomly chooses $TAG_i \in \{0, 1\}^l$ and returns it to \mathcal{A} ; meanwhile, it stores the tuple (x_i, TAG_i) into list ℓ_1 .
- (2) H_μ query: \mathcal{C} maintains list ℓ_2 of tuple (message, M). For a request of message, if message contains list ℓ_2 , \mathcal{C} returns the corresponding tuple to \mathcal{A} ; otherwise, \mathcal{C} randomly chooses M and returns it to \mathcal{A} ; meanwhile, it stores the tuple (message, M) into list ℓ_2 .
- (3) H_3 query: \mathcal{C} maintains list ℓ_3 of tuple $(M, s_i^j, c_i, L)_{i \in [n], j \in [u]}$. For a request of M , a set L has n public key-vectors and randomly chooses $s_i^j \in \mathbb{Z}_q$ ($i \in [n]$, $j \in [u]$); if (M, L) contains list ℓ_3 , \mathcal{C} returns the corresponding tuple to \mathcal{A} ; otherwise, \mathcal{C} randomly chooses $c_i \in \mathbb{Z}_q$ and returns it to \mathcal{A} ; meanwhile, it stores the tuple $(M, s_i^j, c_i, L)_{i \in [n], j \in [u]}$ into list ℓ_3 .
- (4) Key query: for a request of x_i , if the tuple (x_i, TAG_i) contains list ℓ_1 , \mathcal{C} performs polynomial reconstruction function to generate $f(\cdot)$ and the secret key

Input: $x, m, f(\cdot), N, \hat{N}$
Output: $PP = \{N, \hat{N}, ss, TAG\}$
(1) Let $\varphi_x = \Omega^{\text{IoM}}(x, m, N)$, $\hat{\varphi}_x = \Omega^{\text{IoM}}(x, m, \hat{N})$.
(2) Call $f(\cdot)$ to obtain
(3) $f(\varphi_x) = (f(\varphi_{x(1)}), \dots, f(\varphi_{x(m)}))$.
(4) for $1 \leq i \leq m$ do
(5) Compute $ss_i = \hat{\varphi}_{x(i)} \oplus f(\varphi_{x(i)})$,
(6) $\text{tag}_i = H(\varphi_{x(i)} \| ss_i \| \hat{\varphi}_{x(i)})$.
(7) end
(8) Let $ss = (s_1, \dots, s_m)$, $TAG = (\text{tag}_1, \dots, \text{tag}_m)$.
(9) return $PP = \{N, \hat{N}, ss, TAG\}$.

ALGORITHM 2: Setup (x).

Input: $x_t, u, m, G, U, PP = \{N, \hat{N}, ss, TAG\}$.
Output: PK .
(1) Compute $\varphi_{x_t} = \Omega^{\text{IoM}}(x_t, m, N)$, $\hat{\varphi}_{x_t} = \Omega^{\text{IoM}}(x_t, m, \hat{N})$.
(2) Let $U = \emptyset$.
(3) for $1 \leq i \leq m$ do
(4) Compute $\text{tag}_i' = H(\varphi_{x_t(i)} \| ss_i \| \hat{\varphi}_{x_t(i)})$.
(5) if $\text{tag}_i' == \text{tag}_i$ then
(6) Compute $f(\varphi_{x_t(i)})' = ss_i \oplus \hat{\varphi}_{x_t(i)}$
(7) if $(\varphi_{x_t(i)}, f(\varphi_{x_t(i)})') \notin U$ then
(8) Let $U = U \cup (\varphi_{x_t(i)}, f(\varphi_{x_t(i)})')$
(9) end
(10) end
(11) end
(12) Let $t = |U|$.
(13) if $t \geq k$ then
(14) Call function $\text{lagrange}(\cdot)(0)$ to obtain y
(15) Run $\text{DisThreshold}(y, u-1, u)$ to obtain y
(16) for $1 \leq i \leq u$ do
(17) Compute $PK_i = y_i G$.
(18) end
(19) end
(20) return $PK = \{PK_1, \dots, PK_u\}, y$.

ALGORITHM 3: KeyGen (x_t).

Input: message $\in \{0, 1\}^l$, $g, h, G \in \mathbb{G}$, $R \in \mathbb{Z}_q^*$, $L = \{PK_1, \dots, PK_n\}$ ($PK_i = (PK_i^1, \dots, PK_i^u)$), y .
Output: $\Omega(M) = (I_j, c_1, s_i^j, C_R, M, R_t)_{i \in [n], j \in [u]}$.
(1) Let $H_1(\text{message}) = \mu$.
(2) Compute $C = \mu g + Rh$, $C_R = R_\mu g + R_R h$ and $e = H_2(C, C_R)$ where
(3) $R_\mu, R_R \rightarrow \mathbb{Z}_q^*$.
(4) Compute $M = e\mu + R_\mu$, $R_t = eR + R_R$.
(5) for $1 \leq j \leq u$ do
(6) Choose $a_\pi^j \leftarrow \mathbb{Z}_q$
(7) Compute $I_j = y_j H_p(PK_\pi^j)$; $L_\pi^j = a_\pi^j G$;
(8) $R_\pi^j = a_\pi^j H_p(PK_\pi^j)$;
(9) end
(10) Compute $c_{\pi+1} = H_3(M, L_\pi^1, R_\pi^1, \dots, L_\pi^u, R_\pi^u)$.
(11) for $i = \pi+1, \dots, n, 1, \dots, \pi-1$ do
(12) for $1 \leq j \leq u$ do
(13) Choose $s_i^j \leftarrow \mathbb{Z}_q$

ALGORITHM 4: Continued.

```

(14) Compute  $L_i^j = s_i^j G + c_i PK_i^j$ ;
(15)  $R_i^j = s_i^j H_p(PK_i^j) + c_i I_j$ ;
(16) end
(17) Compute  $c_{i+1} = H_3(M, L_i^1, R_i^1, \dots, L_i^u, R_i^u)$ ;
(18) end
(19) for  $1 \leq j \leq u$  do
(20) Compute  $s_\pi^j = a_j - c_\pi y_j \text{mod } q$ ;
(21) end
(22) return  $\Omega(M) = (I_j, c_1, s_i^j, C_R, M, R)_{i \in [n], j \in [u]}$ .

```

ALGORITHM 4: Sign (message, L , y).

```

Input:  $\Omega(M) = (I_j, c_1, s_i^j, C_R, M, R)_{i \in [n], j \in [u]}$ ,
 $g, h \in \mathbb{G}$ ,  $L = \{PK_1, \dots, PK_n\}$  ( $PK_i = (PK_i^1, \dots, PK_i^u)$ ).
Output: 0 or 1.
(1) Compute  $e = H_2(C, C_R)$ .
(2) if  $eC + C_R = Mg + R'h$  then
(3) for  $1 \leq i \leq n$  do
(4) for  $1 \leq j \leq u$  do
(5) Compute  $L_i^j = s_i^j G + c_i PK_i^j$ ;
(6)  $R_i^j = s_i^j H_p(PK_i^j) + c_i I_j$ ;
(7) end
(8) Compute  $c_{i+1} = H_3(M, L_i^1, R_i^1, \dots, L_i^u, R_i^u)$ ;
(9) end
(10) else
(11) 0
(12) end
(13) if  $c_{n+1} = c_1$  then 1;
(14) else 0;
(15) return 0 or 1.

```

ALGORITHM 5: Verify(Ω , L).

$s_i = f(0)$ and returns it to \mathcal{A} . Meanwhile, \mathcal{C} stores the tuple (x_i, s_i) into the ℓ_4 .

- (5) Sign query: for a request of message, the user's biometric vector x_i , and a set L of n public key-vectors where π is the real secret index, \mathcal{C} generates a signature:
- Check list ℓ_4 , if there exists x_i , the real secret key $s_\pi = y_\pi$.
 - Check list ℓ_2 , if there exists (message, M), the secret message is M .
 - For $j \in [u]$, compute $I_j = y_j H_p(PK_\pi^j)$.
 - For $j \in [u]$, choose $a_\pi^j \leftarrow \mathbb{Z}_q$ and $c_{\pi+1} \leftarrow \mathbb{Z}_q$.
 - When $i = \pi + 1, \dots, n, 1, \dots, \pi - 1$:
For $j \in [u]$, choose $s_i^j \leftarrow \mathbb{Z}_q$; compute $L_i^j = s_i^j G + c_i PK_i^j$; $R_i^j = s_i^j H_p(PK_i^j) + c_i I_j$;
Compute $c_{i+1} = H_3(M, L_i^1, R_i^1, \dots, L_i^u, R_i^u)$, add $(s_i^j, c_{i+1}, C_R, M, R')$ _{$(i \in [n], j \in [u])$} into ℓ_5 . If collision occurs, repeat steps 3 and 5.
 - Output $\Omega(M) = (I_j, c_1, s_i^j, C_R, M, R')$ _{$(i \in [n], j \in [u])$} as a signature. It can be seen from the signing process that $\Omega(M)$ is a valid signature.

Forge: \mathcal{A} outputs a forged signature $\Omega(M^*) = (I_j^*, c_1^*, s_i^{j*}, C_R^*, M^*, R'^*)_{i \in [n], j \in [u]}$. Assume that \mathcal{A} makes no more than $q_1 + q_H + uq_3$ queries to the signing oracles H_1, H, H_3 and $\mathcal{S}\mathcal{O}$.

From Lemma 1, for each successful forgery \mathcal{A} completes with transcript \mathcal{T} , there are $u_{\mathcal{T}}$ queries to H_3 matching the n queries used to verify the signature. Let $X_{i_1}, \dots, X_{i_{u_{\mathcal{T}}}}$ denote the i^{th} forgery and let π be the index for the last verification query; we have $X_{i_\ell} = H_3(M, L_{\pi-1}^1, R_{\pi-1}^1, \dots, L_{\pi-1}^{u_{\mathcal{T}}}, R_{\pi-1}^{u_{\mathcal{T}}})$.

If $i_1 = \ell$, \mathcal{A} produces an attempted forgery Ω that is an (ℓ, π) -forgery. By assumption, there exists (ℓ, π) for giving a successful forgery; $\varepsilon_{\ell, \pi}(\mathcal{T})$ satisfies

$$\begin{aligned} & \geq \frac{1}{u_{\mathcal{T}}(q_1 + q_H + u_{\mathcal{T}}q_3)} \cdot \frac{1}{Q_1(\kappa)} \\ \varepsilon_{\ell, \pi} & \geq \frac{1}{u(q_1 + q_H + uq_3)} \cdot \frac{1}{Q_1(\kappa)} \end{aligned} \quad (5)$$

Then, \mathcal{A} rewinds \mathcal{T} before the ℓ^{th} query and again attempts a forgery on the same set of keys that \mathcal{T}' satisfies

$$\varepsilon_{\ell, \pi}(\mathcal{T}') \geq \frac{1}{u(q_1 + q_H + uq_3)} \cdot \frac{1}{Q_2(\kappa)}, \quad (6)$$

and also a successful forgery, where Q_2 is a polynomial inputting a security parameter κ .

Therefore, the probability that both \mathcal{T} and \mathcal{T}' correspond to verifying forgeries Ω and Ω' is nonnegligible:

$$\varepsilon_{\ell,\pi}(\mathcal{T} \& \mathcal{T}') \geq (\varepsilon_{\ell,\pi}(\mathcal{T}))^2. \quad (7)$$

In the way above, we again obtain a forged signature $\Omega(M'_j) = (I'_j, c'_j, s'_j, C'_R, M', R'')_{i \in [n], j \in [u]}$. For each j , we have $s'_j \neq s_j, c'_j \neq c_j$, and we can solve y'_j as

$$y'_j = \frac{s'_j - s_j}{c'_j - c_j} \text{mod } q, \quad (8)$$

which contradicts the DL assumption. \square

Theorem 2. *In ROM, the proposed scheme is signer-anonymous under the DDH assumption.*

Proof (similar proof to Theorem 8 of [4]). Assume that the DDH problem is hard in the cyclic group generated by \mathbb{G} and suppose there exists a PPT adversary \mathcal{A} against signer ambiguity. After that, given a set L of n public key-vectors of length u , a set of t biometric vectors $\mathcal{D}_t = \{x_1, \dots, x_t\}$, and a valid signature Ω on L signed by a user with respect to a key-vector \overline{PK} such that the corresponding biometric vector \overline{x}_π satisfies $\overline{x}_\pi \notin \mathcal{D}_t$, then, \mathcal{A} can win the game above with probability

$$\Pr[\mathcal{A} \rightarrow \pi] \geq \frac{1}{n-t} + \frac{1}{Q_3(\kappa)} \quad (9)$$

for some polynomial $Q_3(\kappa)$. We certainly can construct a PPT simulator \mathcal{C} which takes as inputs a tuple (G, aG, bG, c_iG) , where $i \in \{0, 1\}$ is randomly chosen and not a priori known to \mathcal{C} , $c_1 = ab$, and c_0 is a random scalar; then \mathcal{C} can output i and solve the DDH problem with probability

$$\Pr[\mathcal{C}(G, aG, bG, c_iG) \rightarrow i] \geq \frac{1}{2} + \frac{1}{Q_4(\kappa)}, \quad (10)$$

for some polynomial $Q_4(\kappa)$.

Inputting scalars a, c , the user's biometric vector x , a set L of n public key-vectors of length u , index π , and message M , we act as follows.

Initialization, Query: it is the same as that in Theorem 1.

Challenge: \mathcal{A} submits a new tuple $(M, x_{i,0}, x_{i,1}, L)$, where the public key-vectors corresponding to $x_{i,0}, x_{i,1}$ are in L . \mathcal{C} flips a coin $b \in \{0, 1\}$ and then returns \mathcal{A} with the signature $\Omega(M, x_{i,b}, L)$.

Guess: \mathcal{A} outputs a bit $b' \in \{0, 1\}$.

Given a tuple (G, aG, bG, c_iG) where a, b are randomly selected scalars, with $c_1 = ab$, c_0 is a random scalar, $i \in \{0, 1\}$, \mathcal{C} takes the following steps to solve the DDH problem with nonnegligible probability.

Firstly, \mathcal{C} grabs a private/public key-vector pair $(\overline{y}, \overline{PK})$ from H_1, H_3 and a random γ and computes $s = a - \gamma y$. Then, \mathcal{C} performs SIMNIZKP [4] with arbitrarily selected key-vectors $\{\overline{PK}_i\}_{i \in [n]}$ such that $\overline{PK} = \overline{PK}_\pi$, $a \rightarrow a$, $c_i \rightarrow c$, some message M , and $\overline{y} \rightarrow \overline{y}$.

If $i = 1$, then $c = ab$, $\log_G aG = \log_{bG} cG = a$; and due to the fact that \mathcal{A} is assumed to be able to find π with nonnegligible probability, then there is a nonnegligible probability over $1/2$ that \mathcal{A} returns 1 (upon which \mathcal{C} returns 1). Meanwhile, if $i = 0$, then \mathcal{A} returns 1 only with probability $1/2$, and so for some nonnegligible probability over $1/2$, \mathcal{C} returns the same value as \mathcal{A} and thus solves the DDH problem for randomly chosen scalars with nonnegligible probability over $1/2$, which is a contradiction. \square

Theorem 3. *In ROM, our scheme satisfies linkability.*

Proof (similar proof to Theorem 7 of [4]). Suppose that a PPT adversary \mathcal{A} can produce two unlinkable signatures (Ω_1, M_1, L_1) and (Ω_2, M_2, L_2) with nonnegligible probability, and they signed with respect to public key-matrices L_1 and L_2 such that there exists a public key PK in both L_1 and L_2 that is negligible.

Suppose to the contrary that Ω_1 and Ω_2 both signed with respect to public key-matrices L_1 and L_2 such that there exists a public key PK in both L_1 and L_2 , then with overwhelming probability, there exists the indexes π and π' for the public keys in Ω_1 and Ω_2 , respectively, such that

$$\begin{aligned} L_\pi^j &= s_\pi^j G + c_\pi PK_\pi^j; R_\pi^j = s_\pi^j H_p(PK_\pi^j) + c_\pi I_j; \\ L_{\pi'}^j &= s_{\pi'}^j G + c_{\pi'} PK_{\pi'}^j; R_{\pi'}^j = s_{\pi'}^j H_p(PK_{\pi'}^j) + c_{\pi'} I_{j'}. \end{aligned} \quad (11)$$

Thus, we have

$$\log_G L_\pi^j = \log_{H_p(PK_\pi^j)} R_\pi^j; \log_G L_{\pi'}^j = \log_{H_p(PK_{\pi'}^j)} R_{\pi'}^j. \quad (12)$$

For I_j and $I_{j'}$, it follows that $I_j = y_j H_p(PK_\pi^j) = y_j H_p(PK)$ and $I_{j'} = y_{j'} H_p(PK)$; the private key is related to the user's biometric vector within $|U| \geq k$. In this way, two signatures Ω_1 and Ω_2 include $I_j = I_{j'}$; since the duplicate key images are rejected, one of them must not verify. \square

Theorem 4. *In ROM, our scheme is a NIZK argument of plaintext knowledge.*

Proof. In the above scheme, we have completeness, computational soundness, and zero-knowledge. Proving procedures is as described in [12]. \square

6. Efficiency Analysis

In this section, we give a brief efficiency comparison among the scheme of ours [4, 10]. The communication complexity of our scheme is almost $\mathcal{O}(u(n+1))$ where n is the number of public key vectors in each set. As shown in [4], the communication complexity of its scheme is also $\mathcal{O}(u(n+1))$. In contrast, it is $\mathcal{O}(n)$ in [10].

As shown in Table 2, the cost of time and communication in our scheme is larger than [4, 10]. However, our scheme constructs a linkable ring signature scheme based on NIZK and SKE protocols to enhance the status of privacy-preserving, which is the vital improvement of our scheme.

TABLE 2: Comparison of time and communication cost.

Ref.	Prover	Verifier	Communication
[4]	$2.1un \cdot \exp_1 + un \cdot H_q$	$2.2un \cdot \exp_1 + un \cdot H_q$	$u \mathbb{G} + H + un \mathbb{Z}_q $
[10]	$0.4(n+1)(u+1) \cdot \exp_1 + (n+1) \cdot p + (n+1) \cdot \exp_T$	$3.2\lambda(n+1) \cdot \exp_q + (1.1\lambda + 0.4u + 4)(n+1) \cdot \exp + 3(n+1) \cdot p + 2.2(n+1) \cdot \exp_T$	$(\lambda+5)(n+1) \mathbb{G} + (3\lambda+6)(n+1) \mathbb{Z}_p + 3\lambda(n+1) \mathbb{Z}_q + (n+1) \mathbb{G}_T $
Ours	$2.5un \cdot \exp_1 + un \cdot H_q$	$2.7un \cdot \exp_1 + un \cdot H_q$	$(u+1) \mathbb{G} + H + (u+2) \mathbb{Z}_q $

[*] “ n ”: the number of public key-vectors in each set; “ u ”: the length of each public key-vector; “ λ ”: the length of element in \mathbb{Z}_p ; “ \exp_1 ”: an exponentiation operation in group \mathbb{G}_1 ; “ \exp_T ”: an exponentiation operation in group \mathbb{G}_T ; “ \exp_q ”: an exponentiation operation in group $\mathbb{G}_q \subset \mathbb{Z}_p^*$; “ p ”: a bilinear pairing operation; H_p : a map-to-point hash function; $|H|$: the output size of a hash function H ; $|\mathbb{G}_1|$: the length of element in group \mathbb{G}_1 , similarly for $|\mathbb{Z}_p|$, $|\mathbb{Z}_q|$, and $|\mathbb{G}_T|$.

7. Experiments and Discussion

In this section, we present a simple demonstration of our scheme by using *Python*.

7.1. Parameters. In order to get fingerprint vectors, we adopt the idea of fingerprint image conversion to fingerprint vector which is consisted of several sequential stages: getting fingerprint image, preprocessing, and taking attribute values for feature key points. For each user, one of the fingerprint vectors is used for *Setup*, and other fingerprint vectors from the same user are used to retrieve the secret key with the procedure described in Section 5.

In order to preserve the accuracy performance at the same level as in the original fingerprint in [5], we choose the appropriate parameters of SKE algorithm to control FAR, FRR, and EER within a suitable range in our experiments. Meanwhile, these parameters can also get a theoretically favorable level of the computational security to resist the brute-force and false-accept attacks. For all the experiments, we choose $m = 1024$, $q = 251$, $q' = 2^{80}$ for the dataset (FVC 2002 subset DB1, DB2) with $k = 18$.

As for the signing process, we make improvements based on the original Ring CT [4] codes by adding the SKE and NIZK protocols. In this way, we generate a new code for our scheme. Specifically, we use Keccak-256 hash function to generate the secret keys as well as the public keys and set parameters of the elliptic curve as same as Ed25519. We implemented our scheme in Intel Core *i* 52.5 GHz, 4 Gb RAM, MacOS 10.13.6. Each element in \mathbb{Z}_q is represented by 32 bytes.

7.2. Experimental Results. For a small ring size, we repeat the signing and verification process separately with $n = 6, 8, 10, 12$ and $u = 4, 6, 8, 10$. The experimental results are presented in Table 1.

For a relatively large ring size, we repeat the signing and verification process; the experimental results are presented in Figure 3.

As shown in Figure 3, the running time of signature in our scheme increases linearly with the number of group accounts. Meanwhile, the running times of signing and verification process are almost the same. It also can be concluded from the construction of our scheme.

As shown in Table 3, when we use a small ring size in our signature scheme, the simulation experiment can be carried out by using an ordinary personal computer. The result of

experiment may have some abnormal deviations for different computer hardware, but overall it represents the results of our scheme where we run the scheme on our personal computer. This is useful for mobile demonstration in section 7. According to the experimental results above, we can see that our scheme can meet the needs of practical applications.

8. Anonymous Cryptocurrency Transaction Model Based on LRS Scheme

Based on the proposed scheme and demonstration above, we propose a relatively private cryptocurrency model. Based on the experimental results on PC and private cryptocurrency model, we design a mobile simulation interface which means it may have a practical application value in mobile phone.

In the transaction schematics (Figure 4), we suppose that Alice sends money to Bob. In this process, Alice’s anonymity is protected by ring signature and NIZK protocol. One-time address is used to protect Bob’s anonymity. Moreover, we use SKE algorithm to protect secret key on both sides and use linkable tag to prevent double spending.

- (1) During the registration phase (Figure 5(a)), while Alice enters the personal information and password as prompted, she also takes the fingerprint photo at the same time. On the input information above, the system runs the biometric encryption algorithm SKE to encrypt Alice’s password. In other words, the system runs *Setup* (x) algorithm and stores the public helper data PP.
- (2) In the login page (Figure 5(b)), Alice uses the new fingerprint to unlock the interface. On input of the new fingerprint x' (the query biometric vector in the Section 2 part A), the system runs *KeyGen* (x') algorithm to recover the password through the public helper data PP in step 1. Similarly, Bob does the same operations.
- (3) As shown in Figure 5(c), Alice transfers the money to Bob. In addition, for multilayer ring signature scheme, Alice chooses the number n . Note could be indicated in remarks column, if it has. Finally, Alice clicks on the sign button to sign for this transaction. For system, Alice generates a one-time address for Bob, and no one other than Bob (including Alice) can recover the full signature key. Then, the system selects the $n - 1$ addresses to get the equivalent amount of currency. Thus, the system inputs all Alice’s previous transactions into a hash function and obtains the hash value. Using the NIZK algorithm stated above to blind the message M in Figure 4(c), the system runs signature algorithm to obtain $\Omega(M)$. Finally, the system generates a transaction Tx and broadcasts Tx throughout the P2P network (Figure 5(d)).



FIGURE 5: Mobile demonstration interface.

- (4) In the verification phase (Figure 5(e)), everyone in the P2P network can verify our transactions without knowing our plain message.

However, we do not use an authentication mechanism in the registration phase which will reduce the security of cryptocurrency transactions model. To deal with this issue, we consider using the following two-factor or three-factor authentication mechanisms. Stanislaw et al. [23] presented a secure two-factor authentication system based on the possession by the user of a password and a cryptocapable device which can achieve end-to-end security. Qiu et al. [24] put forward a provably secure three-factor protocol for mobile lightweight devices, which can achieve truly three-factor security while providing password change friendliness. Wang et al. [25] advanced a new two-factor authentication mechanism to resolve the various issues arising from user corruption and server compromise. Jiang et al. [26] proposed a cloud-centric three-factor authentication and key agreement protocol integrating passwords, biometrics, and smart cards to ensure secure access to both cloud and autonomous vehicles.

9. Conclusion

In this paper, we construct a linkable ring signature scheme based on NIZK and SKE protocols to enhance the status of privacy-preserving. In our signature scheme, we use the SKE algorithm to protect the secret key. Simultaneously, we also use the NIZK protocol to encrypt the plain message without revealing redundant information of the message in the verification process. With respect to the NIZK part, the NIZK protocol we used satisfies three characteristics: completeness, soundness, and zero-knowledge. At the same time, our scheme holds unforgeability, anonymity, and linkability in ROM. We also demonstrate the practical performance of our scheme on a personal computer. The result provides us with strong confidence in applying our scheme in practice.

However, since we apply the SKE and NIZK protocols to original Ring CT scheme to encrypt the secret key and the message, the running time and communication cost of our scheme have slightly increased. Although our signature

scheme may meet a real need, we will improve our scheme to reduce the running time and communication cost in our future work. In addition, the security of our scheme is based on the hardness of the DL problem, which is vulnerable to quantum attacks. In the future research, we will improve our scheme based on postquantum cryptography.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was partially supported by the Key Program of the Nature Science Foundation of Zhejiang Province of China (no. LZ17F020002) and the National Science Foundation of China (no. 61772166).

References

- [1] p. Koshy, D. Koshy, and P. Mcdaniel, "An analysis of anonymity in bitcoin using P2P network traffic," *Financial Cryptography and Data Security*, vol. 8437, pp. 469–485, 2014.
- [2] E. B. Sasson, A. Chiesa, and C. Garman, "Zerocash: decentralized anonymous payments from bitcoin (extended version)," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy (SP)*, pp. 459–474, IEEE, Berkeley, CA, USA, May 2014.
- [3] B. S. Eli, C. Alessandro, T. Eran, and M. Virza, "Succinct non-interactive zero knowledge for a von Neumann architecture," in *Proceedings of the 23rd USENIX conference on Security Symposium (SEC'14)*, pp. 781–796, USENIX, San Deigo, CA, August 2014.
- [4] S. Noether, *Ring Signature Confidential Transactions for Monero*, Cryptology ePrint Archive, 2015, <https://eprint.iacr.org/>.
- [5] Y.-L. Lai, J. Y. Hwang, Z. Jin, S. Kim, S. Cho, and A. B. J. Teoh, "Symmetric keyring encryption scheme for biometric cryptosystem," *Information Sciences*, vol. 502, pp. 492–509, 2019.
- [6] D. Chang, S. Garg, M. Hasan, and S. Mishra, "Cancelable multi-biometric approach using fuzzy extractor and novel bit-wise encryption," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3152–3167, 2020.
- [7] J. K. Liu, V. K. Wei, and D. S. Wong, "Linkable spontaneous anonymous group signature for ad hoc groups," *Information Security and Privacy*, vol. 3108, pp. 325–335, 2004.
- [8] M. Franklin and H. Zhang, "A framework for unique ring signatures," *Financial Cryptography and Data Security*, vol. 7859, pp. 162–170, 2012.
- [9] L. Deng, Y. Jiang, and B. Ning, "Identity-based linkable ring signature scheme," *IEEE Access*, vol. 7, pp. 153969–153976, 2019.
- [10] S.-F. Sun, M. H. Au, J. K. Liu, and T. H. Yuen, "RingCT 2.0: a compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency Monero," *Computer Security - ESORICS*, vol. 10493, pp. 456–474, 2017.
- [11] 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*, pp. 103–112, ACM, New York, NY, United States, Jan 1988.
- [12] J. Groth, "Non-interactive zero-knowledge arguments for voting," in *Proceedings of the Applied Cryptography and Network Security, Third International Conference*, pp. 7–10, Springer, UCLA, USA, June 2005.
- [13] C. Gentry, J. Groth, Y. Ishai, C. Peikert, A. Sahai, and A. Smith, "Using fully homomorphic hybrid encryption to minimize non-interactive zero-knowledge proofs," *Journal of Cryptology*, vol. 28, no. 4, pp. 820–843, 2015.
- [14] Y. C. Tsai, R. Tso, and Z. Y. Liu, "An improved non-interactive zero-knowledge range proof for decentralized applications," in *Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, pp. 129–134, IEEE, Newark, CA, USA, April 2019.
- [15] A. Juels and M. Sudan, "A fuzzy vault scheme," in *Proceedings of the IEEE International Symposium on Information Theory*, p. 408, IEEE, Lausanne, Switzerland, July 2002.
- [16] M. Osadchy and O. Dunkelmann, "It is all in the system's parameters: privacy and security issues in transforming biometric raw data into binary strings," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 5, pp. 796–804, 2019.
- [17] H. D. Dung, S. Willy, and T. H. T. Nguyen, "A multivariate blind ring signature scheme," *The Computer Journal*, vol. 8, pp. 1194–1202, 2019.
- [18] H. Q. Le, D. H. Duong, and W. Susilo, "A blind ring signature based on the short integer solution problem," *Information Security Applications*, vol. 11897, pp. 92–111, 2020.
- [19] Z. Jin, J. Y. Hwang, Y.-L. Lai, S. Kim, and A. B. J. Teoh, "Ranking-based locality sensitive hashing-enabled cancelable biometrics: index-of-max hashing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 2, pp. 393–407, 2018.
- [20] S. Agrawal, C. Ganesh, and P. Mohassel, "Non-interactive zero-knowledge proofs for composite statements," *Lecture Notes in Computer Science*, vol. 10993, pp. 643–673, 2018.
- [21] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A new (k,n)-Threshold secret sharing scheme and its extension," *Information Security*, vol. 5222, 2012.
- [22] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [23] J. Stanislaw, K. Hugo, S. Maliheh, and S. Nitesh, "Two-factor Authentication with end-to-end password security," *PKC*, vol. 10770, pp. 431–461, 2018.
- [24] S. Qiu, D. Wang, G. Xu, and S. Kumari, "Practical and provably secure three-factor Authentication protocol based on extended chaotic-maps for mobile lightweight devices," *IEEE Transactions on Dependable and Secure Computing*, 2020.
- [25] D. Wang and P. Wang, "Two birds with one stone: two-factor Authentication with security beyond conventional bound," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 708–722, 2016.
- [26] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K.-K. R. Choo, "Unified biometric privacy preserving three-factor Authentication and key agreement for cloud-assisted autonomous vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9390–9401, 2020.

Research Article

Security Analysis and Bypass User Authentication Bound to Device of Windows Hello in the Wild

Ejin Kim ¹ and Hyoung-Kee Choi ²

¹Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea

²College of Software, Sungkyunkwan University, Suwon 16419, Republic of Korea

Correspondence should be addressed to Hyoung-Kee Choi; meosery@skku.edu

Received 19 April 2021; Revised 16 June 2021; Accepted 30 June 2021; Published 23 July 2021

Academic Editor: Ahmad Samer Wazan

Copyright © 2021 Ejin Kim and Hyoung-Kee Choi. 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.

Windows Hello is a Fast Identity Online- (FIDO-) based new login system for Windows 10, which provides a single sign-on (SSO) service to diverse online applications. Hardware protection is essential for Windows Hello's security. This paper aims to examine the security of Windows Hello on a device where hardware protection is unavailable. We present the first detailed analysis of Windows Hello's security. The results show that, on a hardware-unsupported device, the authentication data for Windows Hello is not properly protected. We propose a migration attack to compromise Windows Hello's security. In the proposed attack, an attacker extracts authentication data from a device to impersonate a victim in his or her Microsoft online account. We consider the possibility of such an attack to be serious and harmful to our society and demand immediate attention for remediation.

1. Introduction

A new standard, Fast Identity Online version two (FIDO2), provides an alternative to password-based authentication by accommodating high-level yet easy-to-use security for user validation [1]. While a password authenticates a user based on what the user knows, FIDO2 is based on who the user is and what the user has. FIDO2 provides strong, attested, asymmetric public key-based credentials for the authentication of users. FIDO2 authenticates a user by verifying a private key with a public key registered in the FIDO server at the time of initial login. The private key is encrypted with FIDO credentials and never leaves the user's device. FIDO credentials with an asymmetric key pair are higher entropy than text-based passwords and are only valid within the user's device.

Microsoft has adapted FIDO2 and implemented it as Windows Hello in Windows 10 [2]. Windows Hello not only provides single sign-on services for device login but also extends the service to web browsers and desktop applications. Microsoft recommends using Windows Hello on devices that are equipped with the onboard Trusted Platform

Module (TPM) [3]. Windows Hello's security hinges quite heavily on device dependency, which means that even if login credentials are leaked, those login credentials are of no use on other devices and, hence, the accounts are safe. This is because a private key plays a main role in authentication and the login credentials are used to encrypt the private key. Simply transferring the login credentials between devices does not compromise a victim's account. An adversary is required to acquire both the login credentials and the private key.

The private key never leaves a device once it is set for an individual user, which is done at the time of account creation. Furthermore, the private key can be saved in the cryptographic hardware, the so-called TPM, and security data are unavailable outside the TPM. Windows Hello is most secure when authentication data is stored in a TPM. However, TPMs have only been actively deployed very recently [4]. Currently, many Windows devices, especially desktops and servers, run without missing the TPM.

In this paper, we evaluate the security of Windows Hello on a hardware-unsupported device by examining how difficult it is to break the device dependency. We analyze, in

detail, how Windows Hello works to identify authentication data on a device and how that data is processed by the operating system for user authentication. Based on the detailed analysis, we propose a migration attack. The attack allows authentication data to be transferred between devices without being limited by device dependency. Anyone, including adversaries, can impersonate a victim in Microsoft's online services. In addition, we built a migration tool, which is available as an open-source solution. This tool extracts essential data used for authentication from a victim's device and, if necessary, disarms any protection applied to the data. The tool then reinstates the same protections on the data to fit it to an adversary's device. Despite Microsoft's claims to the contrary, the device dependency is not as secure as we would hope on hardware-unprotected devices.

The remainder of this paper is structured as follows. Section 2 contains background information relevant to FIDO2 security, including how Windows Hello adapts FIDO2. Section 3 introduces an overview of the proposed attack, the Windows Hello migration attack. Sections 4–6 give a step-by-step explanation of the proposed attack based on how Windows Hello works. Section 7 evaluates the impact and feasibility of our attack in the wild. Section 8 presents related works regarding the security of FIDO2 and Windows Hello. Section 9 concludes this paper.

2. Passwordless Authentication

The password is the most popular user authentication method; password protection is employed by many web services and operating systems, including Microsoft Windows. Password-based authentication does not achieve the extremely high level of security that we would like, because passwords can be stolen or hijacked [5]. Thus, passwordless authentication was proposed in an effort to compensate for the weaknesses of passwords. Fast IDentity Online version two (FIDO2) is an alternative to password-based authentication that provides high-level yet easy-to-use security for user validation [1].

2.1. Passwordless FIDO2 Standard Authentication. FIDO2 uses a suite of user credentials, such as biometrics, personal identification numbers (PINs), and external devices like mobile devices and Universal Serial Bus (USB) devices, to support passwordless login. It relieves users of the need to remember long passwords and to expose their credentials in public. FIDO2 was quickly adopted by many web services and operating systems. Windows, Android, and major browsers such as Edge and Chrome now officially support FIDO2 authentication.

FIDO2's security depends on separating the user's credentials in user devices from the FIDO server. The first time a user accesses a FIDO-enabled web service, the user generates an asymmetric key pair for authentication purposes. Once it is encrypted by the user's credentials, the private authentication key is saved in the user's device. The public authentication key is registered on the FIDO server. This key pair is used by the FIDO server to authenticate the user

during each subsequent login attempt. The credentials and the private key never leave the user's device.

FIDO2 serves two main functions: (1) user authentication and (2) device attestation. Figure 1 illustrates the implementation of these main functions in two protocols, the web authentication (WebAuthn) protocol [6] and the Client to Authenticator Protocol (CTAP) [7]. The web authentication protocol defines an authentication procedure between a user and a web client, such as a browser, and an attestation procedure between a user device and a FIDO server. A FIDO authenticator is a cryptographic entity that authenticates users and protects the asymmetric key pair for attestation. Internal and external authenticators in a user device play a key role in authentication and attestation on behalf of users.

The internal authenticator is responsible for user authentication using PINs or biometrics readings, such as a fingerprint, iris scan, and face recognition. A user may use an external authenticator using Bluetooth low energy (BLE), near-field communication (NFC), and the security key. The CTAP is a protocol that carries messages between a user device and an external authenticator.

A user may have different authenticators in different devices and for different services, which means that the user may use the same credentials in multiple devices and services without sacrificing security. Simply acquiring the credentials does not help an adversary hijack a user's account. This is true because users are uniquely authenticated by a public key in a server and a matching private key that is stored in a device encrypted by the credentials. In order to hijack the user's account in the web service, the attacker must steal the user's device and credentials.

2.2. Microsoft's Adaptation of FIDO2, Windows Hello. We focus on a Microsoft adaptation of FIDO2, called Windows Hello; this is a new user authentication technology first implemented in Microsoft Windows 10. Windows Hello helps users access Microsoft applications and third-party applications without the need for passwords, which can be problematic. Windows Hello authenticates a user using a PIN or biometric readings, called the gesture. The gesture plays the same role as the credentials in FIDO2. Windows Hello uses asymmetric key pairs to authenticate the user and the device to the Microsoft server. The gesture never leaves the device, nor is it exposed to the Microsoft server. Microsoft claims that the user's gesture, even if it is a short PIN, is more secure than a traditional password [8].

Windows Hello officially supports three types of user login: (1) device login with a local account, (2) device login with a Microsoft account, and (3) application login with a Microsoft account. A user may choose either a local account or a Microsoft account. A user with a Microsoft account can access multiple devices with a single account, while with the other types of account a user must have an independent account for each device. Furthermore, the Microsoft account enables users to access a number of services offered by Microsoft or third parties running on a Microsoft service framework. This is a single sign-on (SSO) service. Another

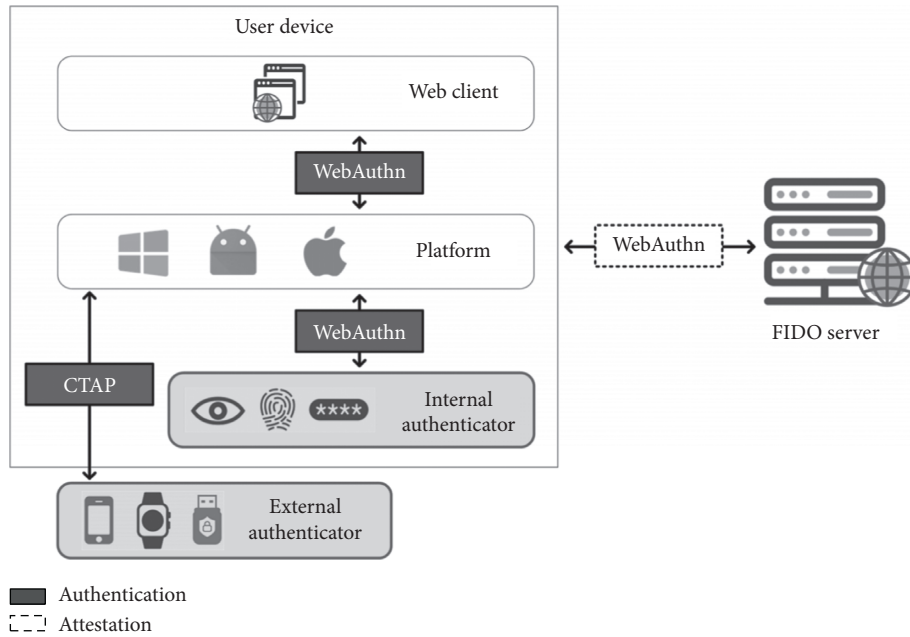


FIGURE 1: FIDO2 works with WebAuthn and CTAP. Through WebAuthn, an internal authenticator in the platform communicates with the FIDO server to authenticate a user. Optionally, the user activates an external authenticator with CTAP using other security devices.

difference between these two types of accounts is that they place information associated with authentication on either a remote server or a local device. In any case, Windows Hello is the default setting for user authentication in Windows 10. With either a local or a Microsoft account, authentication is initiated by a user providing his or her gestures.

When a user registers a Microsoft account on a local device, the use of FIDO's CTAP implementation for mobile applications can enhance the security of the account. If the user activates the two-factor authentication option for the Microsoft account, they log in to the account by entering a security code delivered to the Microsoft authenticator application on the Windows device [9]. After the initial login, the service is used through Windows Hello without additional CTAP authentication.

3. Windows Hello Migration Attack

3.1. Motivation and Goal. We show that an attacker who steals the victim's PIN can access Microsoft services from any device. When Windows Hello is used in combination with single sign-on services, the problem becomes more serious. When the authentication data for Windows Hello is not properly protected, all associated single sign-on services with Windows Hello will be exposed to danger if an attacker steals only the authentication data for Windows Hello.

We first explain how Windows Hello works on general devices. Based on the analysis, we propose a new attack, the Windows Hello migration attack. The attacker's goal is to access the victim's account without authorization. The attacker first obtains the gesture and the authentication data for Windows Hello on the victim's device. Having done so, the attacker can then access the Microsoft account, as well as any resources that are stored on the victim's

device. Furthermore, the attacker can access cloud-based Microsoft applications, such as Office 365 and Microsoft Store and third-party applications. The proposed attack works even if the user has set up two-factor authentication for the Microsoft account. As a result, the proposed attack bypasses all FIDO authentication methods adopted by Microsoft.

3.2. Attack Model. A victim owns a Microsoft account running on a Windows device. The victim has enabled two-factor authentication using the Microsoft authenticator application. Our attack initiates on the device once the victim completes the registration at Microsoft for the use of Windows Hello.

Windows Hello's authentication data are stored in the hard disk and protected by Windows. The Access Control List (ACL) in Windows restricts users' and services' access to authentication data [10]. Only users and services with a valid security identifier (SID) are granted access to authentication data [11]. The SID is given to those who have an administrator's privilege. However, these multiple layers of security can be compromised by the migration attack.

The migration attack presumes the administrator's privilege available to the attack. A number of techniques and programs are available to us and attackers not to believe that our assumption is too optimistic [12, 13]. In one way, the attacker may entice victims to download malware through social engineering like e-mail phishing. In another way, the attacker may remotely exploit arbitrary code execution (ACE) vulnerabilities that existed in software running on the victim's device. The vulnerabilities can be found at the level of applications such as web browsers, services, kernels, and drivers in Windows.

The next step is to bypass a user consent requested by the user account control (UAC) [14]. The attacker attempts to escalate the privilege to gain the administrator's privileges without being noticed by the victim [15–17]. One of the most common ways is to find the credentials of an administrator account required when a normal user wants to use the administrator privilege. The attacker attempts to obtain the existing credentials of the administrator account through storage exploration, memory scraping, and guessing attacks, or add a new account with administrator privileges to the device. In other ways, the attacker seeks ways to set the attacker's process with the high privilege by manipulating permission tokens and policies managed by the operating system. This allows the attacker to modify ownership of the attacker's process and bypass access control. Another commonly used method for privilege escalation is exploiting vulnerabilities in software. The attacker abuses applications or services running with the administrator privilege. The attacker inserts malicious code or library into memory using programming errors at the kernel level.

Privilege escalation vulnerabilities are constantly found, and recently discovered vulnerabilities can be easily found in the Common Vulnerabilities and Exposures (CVE) list [18]. Our study assumes that the attacker already has infiltrated the victim's device and gained the administrator privilege in any way.

The attacker launches the migration attack in four steps, which will be discussed in the remainder of the paper. Figure 2 illustrates the flow of the migration attack. The attacker identifies the data that is used to authenticate the user and the device to the Microsoft server (① in Figure 2). The attacker extracts the authentication data for Windows Hello, which is stored in the victim's device (② in Figure 2). The attacker receives data through the network and calibrates the attacker's device with the victim's authentication data (③ in Figure 2). After the data is migrated, the attacker can access applications with the victim's account on the attacker's device (④ in Figure 2).

4. Authentication Protocol in Windows Hello

The attacker's ultimate goal is to log in to the application with the victim's account. In order to do so, the attacker examines the data that is exchanged between the Microsoft server and a device to validate the user, the account, and the device when the user attempts to log in. To examine the process that occurs when the device communicates with the Microsoft server for the first time, we begin our analysis at the point at which the user creates a local account on a Windows device.

Windows Hello users must perform the three following steps: (1) provision a Windows device and a Microsoft account, (2) set up the gesture, and (3) log in to an application with Windows Hello. After elaborating on all three steps, we will discuss our analysis of Windows Hello authentication security.

4.1. Step 1: Provision of a Windows Device and a Microsoft Account. When a Windows device and a Microsoft account are first provisioned, the device receives two tokens from

the Microsoft server, as shown in Figure 3. The Microsoft server and the user device communicate using the Extensible Markup Language (XML) protocol [19] which encapsulates all data that is transmitted to the network. The server and the device use XML encryption [20] and XML signature [21] technology to ensure the integrity and confidentiality of sensitive information in XML. An XML key K_{XML} which encrypts and signs sensitive data is issued when the local account is created on the Windows device and registered to the server (① and ② in Figure 3). A device token, denoted by T_1 , is an identifier of the XML key. The device transmits the device token with each communication to specify the XML key used for XML encryption and the XML signature.

The other token is an account identifier, denoted by T_2 (④ in Figure 3). As long as T_2 does not expire, the user can access Microsoft services, including registration of Windows Hello, without entering the account password. In this phase, the Microsoft server associates the device token T_1 with the account token T_2 so that the account token can be used together with the associated device token. T_1 plays the role of an identifier for the device and an XML key. Another parameter denoted as *AuthTicket* in ④ in Figure 3 enables the user to sign in to the device with the Microsoft account.

If the user has enabled the two-factor authentication option in the Microsoft account, they need to perform additional authentication with the Microsoft authenticator application when signing in with the Microsoft account (between ③ and ④ in Figure 3). T_2 is a token containing a state in which two-factor authentication has been completed. Subsequent requests using T_2 are not checked for two-factor authentication.

4.2. Step 2: Set Up a Gesture. Figure 4 illustrates the six messages that are exchanged to set up a gesture. For gesture setup, a user verifies themselves to the Microsoft server using T_2 in the second message. If T_1 and T_2 are not associated, the server requests the account password. In the third message, the server returns a renewal of T_2 , denoted by T_2' , along with a new token, T_3 . The new token will be used later, in the fifth message, to confirm the user's account on the device. When the user generates the gesture, an asymmetric key pair is also generated. This key pair is unique to each device, which means that the same user's account in different devices will have different keys.

After the private key is registered, the association between the device token (T_1) and the account token (T_2) is broken up. Instead of the device token, the Microsoft server verifies the association between the account token and the asymmetric key pair. The device token is not involved in authentication; it only works as a key identifier for XML encryption and XML signature.

After being encrypted with the gesture, the private key K_{Pri} is encrypted and saved in a file accessible only by a user with an administrator privilege. The public key K_{Pub} is sent to the Microsoft server along with T_3 in the fifth message in Figure 4. The gesture never leaves the device where it was created.

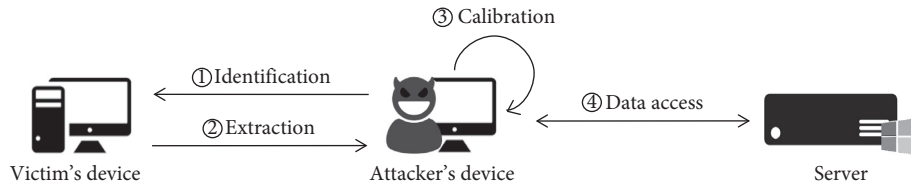


FIGURE 2: Flow of Windows Hello migration attack. An attacker identifies a victim’s authentication data for Windows Hello and extracts it from the victim’s device. The attacker then calibrates it to their device. Finally, the attacker accesses applications with the victim’s account.

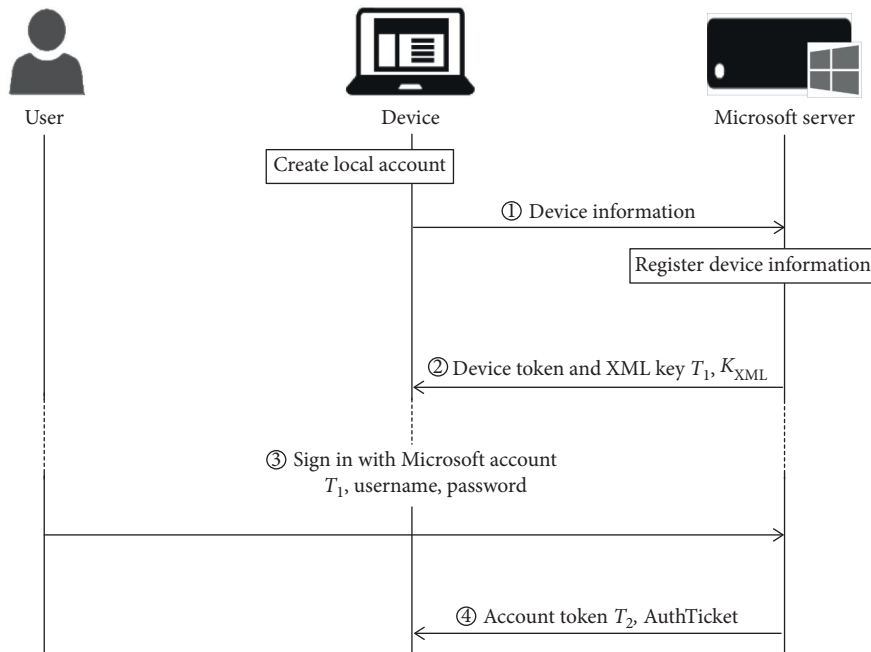


FIGURE 3: Detailed protocol flow: provisioning a Windows device and a Microsoft account. When a user creates a local account in a Windows device and logs in with the Microsoft account, the Microsoft server issues encrypted tokens to identify the device and the account.

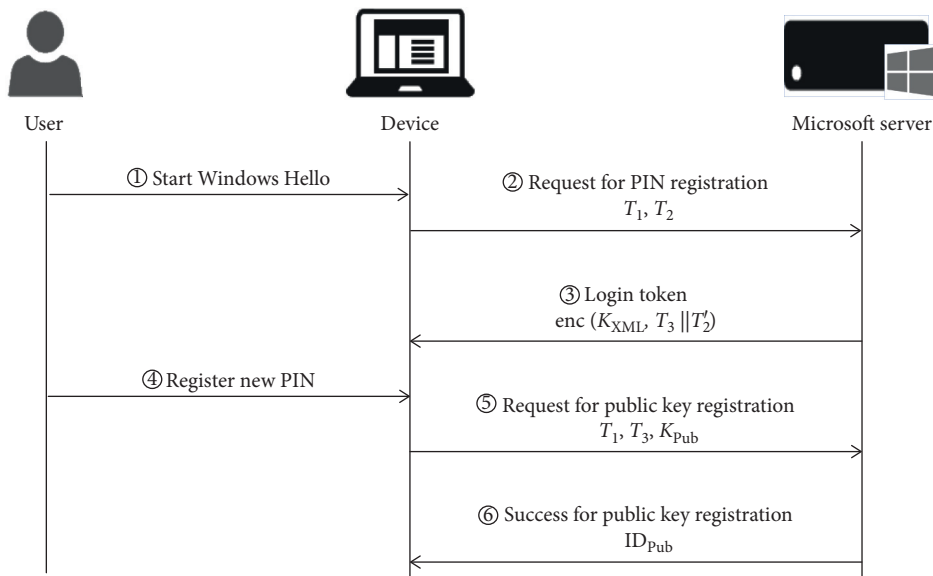


FIGURE 4: Detailed protocol flow: setting up the gesture. Mutual authentication is performed between the user and the server, which involves verifying the user with the device and authenticating the device with the Microsoft server. Through this process, the user’s account and the device are bound.

4.3. *Step 3: Log In to Application with Windows Hello.* At this point, the user's account has been set up and the user is ready to partake in services provided by Microsoft and third parties. Figure 5 shows an application login protocol. Before the user accesses the services, the user must log in to the Microsoft server for authorization. The Microsoft server challenges the user with a random number R in the second message and expects a digital signature of the challenge. The private key is encrypted with the gesture. Only the user with the valid gesture has the private key, and only that person is able to generate the digital signature. The Microsoft server verifies the signature in the fourth message with the corresponding public key.

The server returns an authorization, which is composed of two tokens encrypted with a session key that is generated during the sign-in phase of the Microsoft account and shared between the device and the Microsoft server. The first token, T_4 , is an access token that provides access to a service. The second token, T_2'' , is the second renewal of an account identifier. Once the access token sent in the sixth message is authenticated by the service provider, the user can begin using the service.

This protocol analysis enables us to complete the first stage in the migration attack detailed in Figure 2. In the process of logging in to the application using Windows Hello, the Microsoft server authenticates the user and the device with an account token (T_2) and a private key (K_{Pri}). Those three pieces of information are saved in a device, more specifically in the file system. The attacker needs to identify T_2 and K_{Pri} , which contain information on the format and storage location of the device. The attacker knows that K_{Pri} is encrypted with the gesture. The gestures used to encrypt K_{Pri} include a registered PIN, biometrics, and security keys. We focus on how the PIN is used and how the attacker reveals the PIN in the victim's device.

5. Data Management in Windows Hello

In this section, we discuss the storage and use of the account token and private key and show how the attacker can extract them from the victim's device and calibrate them to fit the attacker's device. After the account setup described in ⑤ in Figure 4, Windows Hello ends up owning three additional pairs of the asymmetric key. These keys have the same structure but are saved in different files. A private key file consists of four elements: (1) RSA key identifier, (2) public key, (3) private key properties such as salt and an iteration count for encrypting or decrypting, and (4) encrypted binaries of the private key.

Windows 10 supports a set of interfaces for handling private keys in a secure manner. These interfaces are defined and declared under a subsystem called the Data Protection Application Programming Interface (DPAPI) [22]. DPAPI protects sensitive data in secure storage locations on the disk by encrypting that data with keys derived from a local password. The *(un)protect* function is the implementation of DPAPI operation. The *(un)protect* function allows only the user who has the password for device login to encrypt or decrypt the data and assess its integrity.

$$KEK^i = SHA512(PBKDF2(PIN \text{ or } seed, salt, iteration)). \quad (1)$$

$$K_{Pri}^i = unprotect(KEK^i, EncPriKey). \quad (2)$$

Figure 6 presents the procedure used to decrypt a private key (K_{Pri}^i) from the private key file. A key encryption key KEK^i is computed from the PIN, as shown in equation (1). The *PBKDF2* function increases the difficulty of a brute-force attack by iterating over cryptographic secure hash functions. The values of salt and iteration used in equation (1) are extracted from private key properties that are decrypted from the private key file with a DPAPI key of a Windows account. The result of the *PBKDF2* function is used to compute a key encryption key KEK^i via the *SHA512* function. The key encryption key decrypts the private keys of the encrypted key file through the *unprotect* function in equation (2).

Figure 7 illustrates the procedure used to decrypt four private keys in the Windows Hello login. Table 1 shows the main functions used in the authentication process, while Table 2 shows where authentication-related data is stored.

5.1. ①, ② *Enter the PIN and Decrypt the Private Keys.* The user-supplied PIN is a secret key that is used for the decryption of the first private key (K_{Pri}^1), which further decrypts *seed* in a key metadata file. *seed* is randomly generated when the gesture is set up, and it is used as a secret key to decrypt the second, third, and fourth private keys in the private key files in equation (1).

5.2. ③-① *Device Login with a Local Account.* Local account users are authenticated with an *EncPwd* in the registry. This is an encrypted local password with a second public key. In a decryption procedure, the second private key (K_{Pri}^2) is derived from the *seed* and, thus, from the user's PIN. This is tantamount to saying that the *EncPwd* is decrypted with the PIN. The output of decryption is used as an input to an authentication process for a login in Windows called the New Technology LAN Manager (NTLM) authentication process [23], where a hash of the user's password is stored. If the two passwords match, authentication succeeds.

5.3. ③-② *Device Login with a Microsoft Account.* Authentication for Microsoft account users is quite similar to that for local account users, except that the third private key (K_{Pri}^3) is used to decrypt an *AuthTicket* in the *CacheData* file. The Microsoft authentication server issued the *AuthTicket* to the user when the Microsoft account was registered. *AuthTicket* is used for the user authentication process with the Microsoft server. The device sends *AuthTicket* to the Microsoft server; if *AuthTicket* is valid, authentication is successful.

5.4. ③-③ *Application Login with Windows Hello.* The Windows Hello service signs in a random challenge value received from the Microsoft server with the fourth private

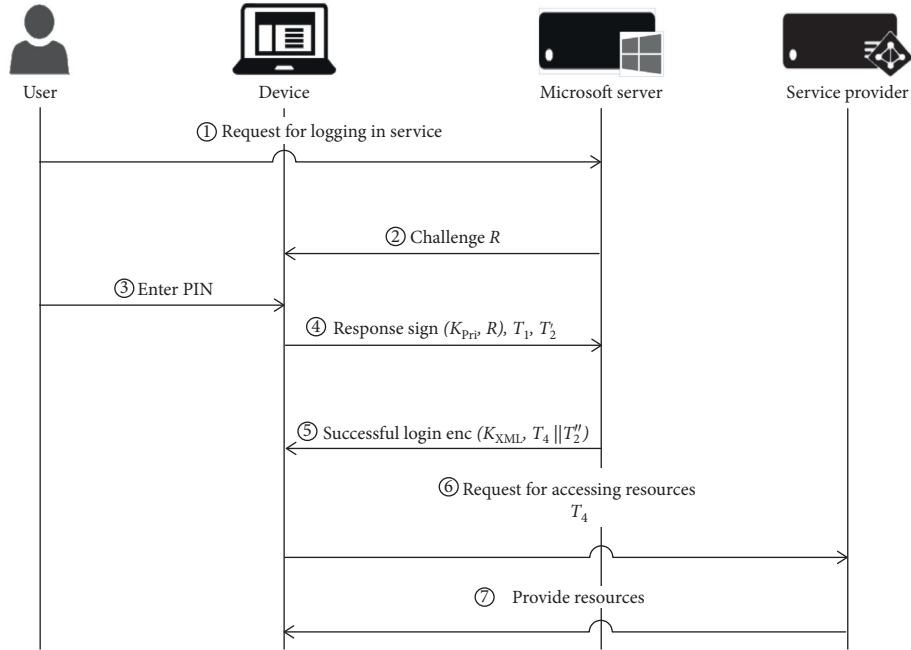


FIGURE 5: Detailed protocol flow: application login with Windows Hello. The user and the device are authenticated to obtain the access token (T_4) for use by the service provider.

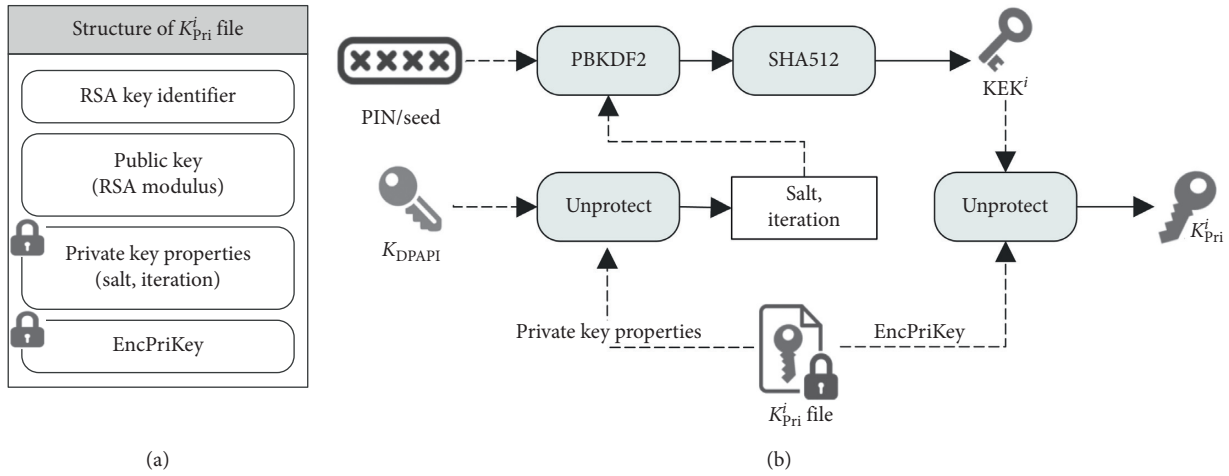


FIGURE 6: Structure and decryption procedure of private key K_{Pri}^i files. To decrypt K_{Pri}^i , the key encryption key KEK^i is computed in the PIN or seed. (a) Structure of key file. (b) Decryption procedure of private keys.

key (K_{Pri}^4). As discussed in Section 3.1, the fourth private key is registered to the Microsoft server when the gesture is set up (© in Figure 4). The Windows Hello service gets T_2 from the credential storage location and sends it to the Microsoft server. After the Microsoft server validates the signed challenge and the token, the Windows Hello service succeeds in logging in to the application and receives the renewed T_2' from the server.

In Windows 10, credential storage is the repositories that encrypt and store security essentials used in the user’s local, network, and web accounts. T_2 , which is issued by the Microsoft server, is stored in credential storage. DPAPI is used to protect the data in credential storage.

6. Details of the Migration Attack

As discussed in the above sections, Windows Hello works based on private keys that are encrypted with the user’s PIN. To migrate the authentication data for Windows Hello, the attacker must first figure out the PIN and then extract the keys by decrypting them with the PIN. The victim’s authentication data is transferred to the attacker’s device, calibrated on the attacker’s device, and used by the attacker to access the victim’s services.

6.1. PIN Cracking. Malware that is injected into a local device can be used to access private key files on the disk. The attacker launches a brute-force guessing attack on these files

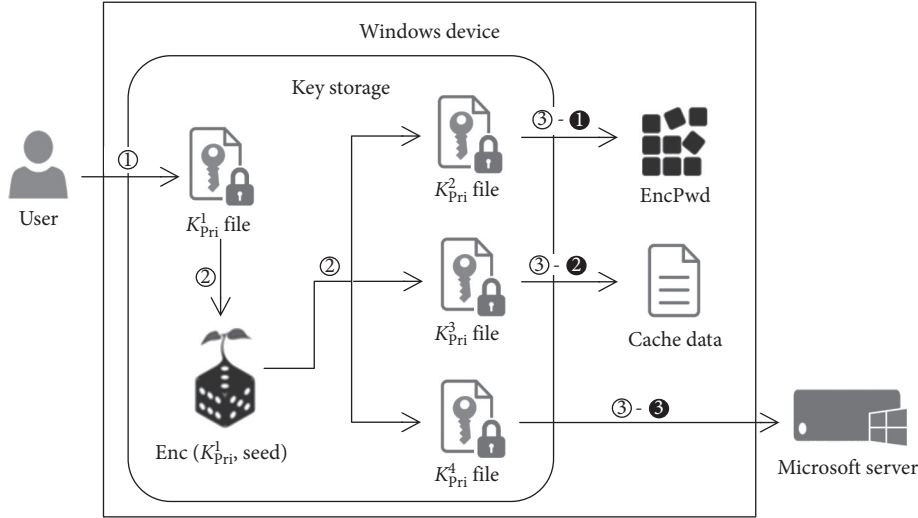


FIGURE 7: Windows Hello login phases showing the roles of the four private keys: ① enter the PIN, ② decrypt private keys, ③-① log in to the device with the local account, ③-② log in to the device with the Microsoft account, and ③-③ log in to the application with Windows Hello.

TABLE 1: Main functions used in the Windows Hello login process. We focus on the role that each function plays in the login process.

Module	Function	Descriptions
<i>BCrypt.dll</i>	<i>BcryptDeriveKeyPBKDF2</i>	Derive a key encryption key through hashing for PIN or <i>seed</i>
<i>Crypt32.dll</i>	<i>CryptUnprotectDataNoUI</i>	Decrypt all private keys with key encryption keys derived from <i>BcryptDeriveKeyPBKDF2</i> function
<i>NCrypt.dll</i>	<i>NcryptOpenKey</i>	Generate key handler based on a key name
<i>CryptSvc.dll</i>	<i>NgcDecryptData</i>	Decrypt the authentication data for a device login
<i>NgcCtnr.dll</i>	<i>DecryptPkcs1</i>	Decrypt <i>seed</i> with the first private key managed by the software key storage provider
	<i>SignHashPkcs1</i>	Sign challenge <i>R</i> with the fourth private key for application login

TABLE 2: Storage of authentication data for Windows Hello.

Data	Path
Private keys	<i>%SystemRoot%\ServiceProfiles\LocalService\AppData\Roaming\Microsoft\Crypto\Keys</i>
Key metadata	<i>%SystemRoot%\ServiceProfiles\LocalService\AppData\Local\Microsoft\Ngc</i>
<i>EncPwd</i>	<i>\HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\LogonUI\NgcPin\Credentials\EncryptedPassword</i>
<i>CacheData</i>	<i>%SystemRoot%\System32\config\systemprofile\AppData\Local\Microsoft\Windows\CloudAPCache\MicrosoftAccount</i>
<i>Credentials</i>	<i>%LOCALAPPDATA%\Microsoft\Credentials</i>

to find the PIN. The PIN is an essential target of the attack as it is used to decrypt private keys. The guessing attack works with offline files, allowing the attack to launch repeatedly without any pauses between attempts and without any limits on the number of attempts. We have developed a tool for PIN cracking. Section 5.2 describes the tool in further detail.

6.2. Migration of Authentication Data. After decrypting the file with the PIN, the attacker extracts the first private key. The attacker also decrypts three other private keys with *seed*. The attacker extracts most authentication data based on the private keys. The extracted data includes the private key

identifiers, *seed*, *CacheData*, registry, and credentials. The data is decrypted from the filesystem and registry on the victim's device and transmitted to the attacker's device.

Some data is protected by the DPAPI mechanism. DPAPI works under security identifiers (SID) [11]. An SID is an identifier that is assigned to identify the user in the system or to grant permissions to users in Windows 10. Data that is encrypted by DPAPI cannot be decrypted without the SID that was used for encryption. When the device needs to use protected data, that data is decrypted using the *unprotect* function executed by the process that is granted by a specific SID.

The Windows Hello service uses DPAPI to protect all private keys with the local authority, *SID_{Local}*. *SID_{Local}*

includes all users who have logged in locally. Data that is encrypted using SID_{Local} is accessible to all users who are currently logged in to the device, without any authentication process. T_2 is also protected by DPAPI and is stored in credential storage. Credentials are encrypted with the LocalSystem authority, SID_{System} , which has full access to the system. The data encrypted using SID_{System} is only accessible to a user with administrator privileges.

The malware running on the victim's device works only at the precise moment the authentication data is extracted. After the authentication data is extracted, the malware no longer needs to run on the victim's device. This makes the attack difficult to detect. Moreover, using DPAPI decryption tools such as Mimikatz [24], the attacker can obtain all authentication data even when the victim's device is not activated and only the disk image is acquired.

The attacker sends the extracted authentication data to the attacker's device. The attacker then calibrates it to be used by the Windows Hello service running on the attacker's device. At this point, the attacker's device is logged in to the attacker's Microsoft account and Windows Hello is enabled. The extracted authentication data replaces the attacker's data. These are also based on the PIN and the first private key. All private keys and credentials are protected by DPAPI using the proper SID on the attacker's device.

Finally, the attacker attempts to log in to the application with the victim's account. All authentication data contains the victim's information, and the Microsoft server recognizes the attacker's device as the victim's device. The attacker enters the victim's PIN and then accesses the application with the victim's account.

7. Evaluation

7.1. Effectiveness of the Migration Attack.

- (1) *Similarities to Hijacking of a Microsoft Account.* A typical session hijacking attack is performed by obtaining session data, such as cookies or tokens, and deceiving the server using that data. The scope of a session hijacking attack is limited to the web services that use the hijacked session data; after the session expires, the attack will not be maintained.

On the other hand, in a Windows Hello migration attack, the victim's authentication data is transferred to the Microsoft server and used to impersonate the victim. Thus, this attack is similar to stealing a Microsoft account. This attack can lead to the hijacking of all web services that are accessible through Windows Hello. Several apps, such as Store, OneDrive, Office, Skype, and Outlook, deal with sensitive information that could be a problem if it is leaked. As the number of applications that are accessible through Windows Hello is expected to continue to increase in the future, the ripple effects of such an attack are expected to increase as well.

- (2) *Stealth.* The victim may recognize that their authentication data for Windows Hello login has been leaked and may make changes or remove the PIN

from the affected device. However, the attacker can still access Windows Hello applications using the victim's account. This is possible because the Windows Hello service does not communicate with the Microsoft server when the PIN is removed or changed.

- (3) *Bypassing Two-Factor Authentication.* The migration attack appears valid even if the user has activated Microsoft authenticator. The tokens that are obtained through the migration attack were previously approved by Microsoft authenticator. If the attacker uses Windows Hello after migrating the victim's authentication data to the attacker's device, the victim's mobile device does not receive an approval notification or alert. As a result, it is possible to steal the victim's Microsoft account using only the Windows Hello authentication data by bypassing two-factor authentication.
- (4) *Privilege Escalation for Changing Login Password.* The attacker cannot steal the login credentials that are used to log in to a Windows 10 device with the Microsoft account, even if the attacker has administrator privileges on the device. In Windows 10, a user with administrator privileges can change the passwords of other users who log in to the local account. However, the user cannot change the password of a device that is logged in to the Microsoft account. The password for the Microsoft account cannot be found on the victim's device. However, the attacker can access the PIN and the authentication data that are used in lieu of a password in Windows Hello.

7.2. Efficiency for PIN Cracking. Microsoft introduced a secure password-based key derivation function (PBKDF) to protect encrypted key files from brute-force attacks [25]. This function mitigates a vulnerability to the brute-force attack by increasing the computational cost of generating encryption and decryption keys. The PBKDF2 function in Windows Hello, which is used when generating a key encryption key KEK^1 for an encrypted key file, increases the time taken by a brute-force attack by repeating the hash function 10,000 times. This makes it difficult to find the PIN within a realistic time.

We measured the feasibility of using a PIN-cracking tool to find the PIN through brute-force attack. The tool takes about one second to decrypt an encrypted key file 100 times with a single-core processor. Table 3 shows the amount of time it takes a brute-force attack to crack a PIN of four to eight digits. For a six-digit PIN, the tool can try every possible combination within three hours. With a multicore processor or Graphics Processing Units (GPUs), it might be possible to hijack PINs with less than ten digits in a realistic timeframe.

One of the reasons that it is possible to crack a PIN is that it is a numeric password. It has a minimum of four digits to a maximum of 20 digits. In PIN-based authentication systems, about 99 percent of users use PINs within ten digits [26].

TABLE 3: Brute-force attack duration according to PIN length.

# of digits	4	5	6	7	8
Time	1.7 min	17 min	2.8 hours	1.1 days	11 days

Some PINs mix letters and special characters with digits, but this compromises convenience, which is one of the greatest advantages of a PIN. With a complex PIN, the user experiences the same usability issues as they do with password authentication.

If the attacker performs a brute-force attack on an online system, the attack is restricted to a certain period. This is because login attempts cannot be performed more than a predefined number of times. The proposed attack is performed by an attacker who holds an encrypted key file offline. Unlike an online brute-force attack, there is no limit to the number of attack attempts that can be performed. Therefore, this is a suitable environment for a brute-force attack.

7.3. Target Application. We performed case studies on applications for which a victim’s account can be stolen through a migration attack. The target applications support a single sign-in service, which, after a user logs in with a Microsoft account on a Windows device, allows automatic logins without additional account credentials. An attacker can access most services used after the victim logs in for target applications:

- (1) *Office 365.* Office 365 is a set of cloud-based document-sharing services provided by Microsoft. Office 365 contains popular document editors such as PowerPoint, Excel, Word, and OneNote. All documents created through Office 365 applications are shared to the cloud via OneDrive. OneDrive provides cloud storage for not only documents created through the Office 365 application but also user data. An attacker who accesses the Office 365 service through the migration attack obtains all data stored in the victim’s OneDrive. The attacker can steal or modify data on OneDrive. Also, by uploading new files such as malware files, the attacker can take over any of the victim’s other devices that are synchronized.
- (2) *Microsoft Store.* Microsoft Store is an online market for purchasing applications that are available on Windows desktops. An attacker who accesses Microsoft Store through a migration attack can see the victim’s wish list and library of previous purchases. The attacker can redownload the purchased applications. If payment information was added to the victim’s account, the attacker can purchase other applications with that payment method.
- (3) *Edge Browser.* Edge browser is a web browser that is provided by default in Windows. Edge browser provides a synchronization service that maintains browser settings across different devices and operating systems. An attacker who accesses the Edge

browser through a migration attack can steal website passwords, history, and payment information.

- (4) *Single Sign-On Website.* An attacker can access websites that allow single sign-on login using a victim’s Microsoft account. When accessing such websites through a web browser, a user can sign in with Windows Hello by selecting the option to log in with a Microsoft account. The option for logging in to websites using Windows Hello is supported by all major browsers such as Google Chrome, Microsoft Edge, and Mozilla Firefox. The attacker can impersonate the victim to access the services provided by the websites.

8. Related Works

FIDO2 is a successor to FIDO1 [27]. FIDO2 extends FIDO1’s coverage from mobile devices to web services and various operating systems. Along the same lines as FIDO2, FIDO1 uses an asymmetric key pair based on trusted platform computing to authenticate a user without exposing the user’s password. FIDO1 provides two subprotocols: (1) the Universal Authentication Framework (UAF) protocol, which plays a similar role to that of FIDO2’s web authentication protocol, and (2) the Universal Second Factor (U2F) protocol, which plays a role similar to that of FIDO2’s CTAP. Most security threats in FIDO1, which operates under the same conceptual framework as FIDO2, can be applied equally to FIDO2. Along with focusing on the security analysis and possible attacks on FIDO2, we also investigated related works about FIDO1 which may be relevant to FIDO2 security.

Table 4 summarizes previous works that are relevant to our study. We were interested the protocol, analysis method, and attack model for security analysis on which related works are focused. Most studies focus on the formal or informal analysis of protocol design. A few studies on FIDO implementation were conducted for the Android platform and U2F-supported websites. Attack models can be divided into six categories: (1) malicious or curious server, (2) a network attacker who can intercept, forge, or send all messages, (3) a web attacker who intervenes in the protocol via the web as a third party not directly participating in the authentication, (4) a local attacker who has full control of a Trusted Execution Environment- (TEE-) unsupported user device, (5) a local attacker who has full control of a TEE-supported user device, and (6) a user who wrongly performs some actions or ignores security warnings. The remainder of this section summarizes what threats each study derives from which attack models for the analysis target.

First, informal analyses of the UAF protocol were conducted. Hu and Zhang [28] pointed out the problem of unauthenticated software communication with the FIDO authenticator, which allows the malware to impersonate benign software in order to hijack credentials within the FIDO authenticator. Similarly, Panos [29] used informal analysis to present attack vectors that exist in the UAF protocol. This study points to a major potential problem: a

TABLE 4: Comparison of security properties of related works. Our focus is on the threats and attack scenarios presented by the studies and whether the attacks are feasible. We investigated the protocol, analysis method, analysis target, and attack model targeted for previous works.

Item	Property	[28]	[29]	[30]	[31]	[32]	[33]	[34]	[35]	[36]	[37]	[38]	[39]
Protocol	UAF/WebAuthn	•	•	•	•					•		•	•
	U2F/CTAP					•	•	•	•	•	•		
Analysis target	Design	•	•	•	•	•	•	•	•	•			•
	Implementation										•	•	•
Analysis method	Formal			•	•				•	•			•
	Informal	•	•			•	•	•			•	•	
Attack model	Server attacker			•	•	•	•	•			•		
	Network attacker				•			•		•	•		
	Web attacker			•			•						
	Local attacker (without TEE)	•	•		•	•	•		•	•	•		
	Local attacker (with TEE)		•		•	•		•				•	•
	Human error								•				

trusted computing platform that manages FIDO credentials can be replicated or bypassed by an authorized attacker.

Several works use formal analysis to examine potential security threats in the design of the UAF or WebAuthn protocol. Guirant and Halpin [30] discussed the privacy issue involved with identifying a user in a different server when one authenticator is used in two web servers simultaneously. Xu [31] focused on the environment in which the UAF protocol is based, the Trusted Execution Environment (TEE), which is used, for example, in the TPM chip. Alaca and Oorschot [32] warned of possible threats in which authentication data is extracted from a disk or memory by malware on user devices.

FIDO U2F has also been analyzed in both informal and formal manners. Chong [33] presented an analysis of the U2F protocol, including a discussion on possible attacks. The U2F protocol uses the channel ID and origin site as challenge values to prevent password reuse, phishing, and Man-in-the-Middle (MITM) attacks. However, if the FIDO client is compromised, the protocol remains vulnerable to MITM and Denial of Service (DoS) attacks. An attack vector that compromises the FIDO client is theoretically similar to our attack vector, but previous works do not provide details of the attack scenarios. Pereira et al. [34] and Jacomme and Kremer [35] modeled the FIDO authentication process and formally analyzed the protocol phase by phase using the ProVerif tool. They validated security of various threat scenarios that may occur in practical environments. They discussed an attack model that is related to our work in which a victim’s platform is compromised and an attacker can control FIDO1 authentication with the victim’s permissions. Barbosa [36] performed a formal analysis of both WebAuthn and CTAP, where cryptographic components required for the use of FIDO2 securely through a provable security analysis were defined.

Few studies have investigated the implementations of FIDO. Patat and Sabt [37] targeted YubiKey [40], which is an implementation of the U2F protocol. They focused on a feature called “remember me,” which allows a device to log in with only a password after initial U2F authentication to make use of U2F authentication less of a hassle. They

demonstrated the feasibility of exploiting the feature by logging in to the Facebook website from a device that has not performed the U2F authentication. Li et al. [38] analyzed the UAF protocol implemented on the Android platform. Based on the analysis, they proposed a rebinding attack, which binds the victim’s FIDO identifier to the attacker’s FIDO authenticator and accesses the service with the victim’s account. This study points out that the feasibility of the rebinding attack is due to the lack of proper authentication between the FIDO authenticator and the FIDO server. Feng [39] formalized the security model and protocol for various scenarios and then developed an automated verifier which performs security analysis and identifies design flaws according to the security assumptions and goals. Unlike the present paper, that work does not focus on how authentication data is managed on authenticators.

9. Conclusion

In this work, we presented a detailed analysis of Windows Hello implementation on Microsoft Windows 10. This work provided the first empirical study of the security of the Windows login system recommended by Microsoft for next-generation devices, the FIDO2 web authentication protocol. We explore the security issues, which are mainly due to the differences between the ideal environment, in which Windows Hello is completely secure, and the actual operating environment. Based on our analysis of Windows Hello authentication, we propose the Windows Hello migration attack to prove the feasibility of our attack scenario. We hope this paper will help researchers reconsider the implementation environment of the FIDO2 authentication protocol.

Data Availability

No data were used to support this study.

Conflicts of Interest

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

Acknowledgments

This work was supported by the Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean Government (MSIT) (no. 2019-0-01343, Regional Strategic Industry Convergence Security Core Talent Training Business).

References

- [1] FIDO Alliance, “FIDO2: WebAuthn & CTAP,” 2018, [Online]. Available: <https://fidoalliance.org/fido2/>.
- [2] M. Docs, “Windows hello for business overview,” [Online]. Available: 2020, <https://docs.microsoft.com/windows/security/identity-protection/hello-for-business/hello-overview>.
- [3] M. Docs, “Trusted platform module technology overview,” 2018, [Online]. Available: <https://docs.microsoft.com/windows/security/information-protection/tpm/trusted-platfor-module-overview>.
- [4] K. Mayes, “An introduction to smart cards,” *Smart Cards, Tokens, Security and Applications*, Tokens, Security and Applications, London, UK, 2017.
- [5] S. W. Shah and S. S. Kanhere, “Recent trends in user authentication - a survey,” *IEEE Access*, vol. 7, no. 1, pp. 112505–112519, 2019.
- [6] World Wide Web Consortium (W3C) and W. Authentication, “An API for accessing public key credentials level 2,” W3C Candidate Recommendation, 2020, [Online]. Available: <https://www.w3.org/TR/webauthn/>.
- [7] FIDO Alliance, “Client to authenticator protocol (CTAP) proposed standard,” 2019, [Online]. Available: <https://fidoalliance.org/specs/fido-v2.0-ps-20190130/fido-client-toauthenticator-protocol-v2.0-ps-20190130.pdf>.
- [8] M. Docs, “Why a pin is better than a password,” 2017, [Online]. Available: <https://docs.microsoft.com/windows/security/identity-protection/hello-for-business/hello-why-pin-is-better-than-password>.
- [9] M. Docs, “Enable passwordless sign-in with the microsoft authenticator app,” 2020, [Online]. Available: <https://docs.microsoft.com/azure/active-directory/authentication/howto-authentication-passwordless-phone>.
- [10] M. Docs, “Access control: understanding windows file and registry permissions,” 2019, [Online]. Available: <https://docs.microsoft.com/en-us/archive/msdn-magazine/2008/november/access-control-understanding-windows-file-and-registry-permissions>.
- [11] M. Docs, “Security identifiers,” 2017, [Online]. Available: <https://docs.microsoft.com/windows/security/identity-protection/access-control/security-identifiers>.
- [12] G. Hull, H. John, and B. Arief, “Ransomware deployment methods and analysis: views from a predictive model and human responses,” *Crime Science*, vol. 8, no. 2, pp. 1–22, 2019.
- [13] S. Sibi Chakkaravarthy, D. Sangeetha, and V. Vaidehi, “A survey on malware analysis and mitigation techniques,” *Computer Science Review*, vol. 32, no. 1, pp. 1–23, 2019.
- [14] M. Docs, “How user account control works,” 2018, [Online]. Available: <https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works>.
- [15] K. Oosthoek and C. Doerr, “SoK: ATT&CK techniques and trends in Windows malware,” in *Proceedings of 15th International Conference On Security And Privacy In Communication Systems (SecureComm)*, Orlando, FL, USA, October 2019.
- [16] M. J. Haber, “Privilege escalation,” *Handbook of Privileged Attack Vectors: Building Effective Cyber-Defense Strategies To Protect Organizations*, 2020.
- [17] A. T. T. Mitre, “C. K., “Privilege escalation” 2021, [Online]. Available: <https://attack.mitre.org/tactics/TA0004/>.
- [18] CVE, “Common vulnerabilities and Exposures list,” 2021, [Online]. Available: <https://cve.mitre.org/cve/>.
- [19] Word Wide Web Consortium (W3C), “XML protocol activity”, W3C’s architecture domain,” 2000, [Online]. Available: <https://www.w3.org/2000/xp/>.
- [20] Word Wide Web Consortium (W3C), “XML encryption syntax and processing version 1.1,” W3C Recommendation, 2013, [Online]. Available: <https://www.w3.org/TR/2013/REC-xmlenc-core1-20130411/>.
- [21] Word Wide Web Consortium (W3C), “XML signature syntax and processing version 1.1,” W3C Recommendation, [Online]. Available: 2013, <https://www.w3.org/TR/xmlsig-core1/>.
- [22] E. Burzstein and J. Picod, “Recovering Windows secrets and EFS certificates offline,” in *Proceedings of 4th USENIX Workshop on Offensive Technologies (WOOT)*, Washington, DC, USA, August 2010.
- [23] M. Docs, “Microsoft NTLM,” 2018, [Online]. Available: <https://docs.microsoft.com/windows/win32/secauthn/microsoft-ntlm>.
- [24] B. Delpy, “Mimikatz,” 2020, [Online]. Available: <https://github.com/gentilkiwi/mimikatz>.
- [25] B. Kaliski, “PKCS #5: password-based cryptography specification version 2.0,” *IETF RFC*, vol. 2898, 2000.
- [26] DataGenetics, “Distribution of all-numeric passwords based on length,” 2012, [Online]. Available: <http://www.datagenetics.com/blog/september32012/>.
- [27] FIDO Alliance, “Specifications overview,” 2014, [Online]. Available: <https://fidoalliance.org/specifications/>.
- [28] K. Hu and Z. Zhang, “Security analysis of an attractive online authentication standard: FIDO UAF protocol,” *China Communications*, vol. 13, no. 12, pp. 189–198, 2016.
- [29] C. Panos et al., “A security evaluation of FIDO’s UAF protocol in mobile and embedded devices,” in *Proceedings of 28th International Tyrrhenian Workshop On Digital Communication*, Palermo, Italy, September 2017.
- [30] I. Guirant and H. Halpin, “Formal verification of the web authentication protocol,” in *Proceedings of 5th Annual Symposium And Bootcamp On Hot Topics In the Science Of Security (HoTSoS)*, Raleigh, NC, USA, April 2018.
- [31] S. Xu et al., “A symbolic model for systematically analyzing TEE-based protocols,” in *Proceedings of 22nd International Conference On Information And Communications Security (ICICS)*, Copenhagen, Denmark, August 2020.
- [32] F. Alaca and P. Oorschot, “Comparative analysis and framework evaluating web single sign-on systems,” *ACM Computing Surveys*, vol. 53, no. 5, p. 112, 2020.
- [33] J. Chong, “Breaking FIDO: are exploits in there?” in *Proceedings of the 19th Blackhat*, Las Vegas, NV, USA, August 2016.
- [34] O. Pereira, F. Rochet, and C. Wiedling, “Formal analysis of the FIDO 1. x protocol,” in *Proceedings of 10th International Symposium On Foundations And Practice Of Security, (FPS)*, Nancy, France, October 2017.
- [35] C. Jacomme and S. Kremer, “An extensive formal analysis of multi-factor Authentication protocols,” *ACM Transactions on Privacy and Security*, vol. 24, no. 2, pp. 13–34, 2021.
- [36] M. Barbosa et al., “Provable security analysis of FIDO2,” *Cryptology ePrint Archive Report 2020/756*, 2020 [Online]. Available: <https://eprint.iacr.org/2020/756.pdf>.

- [37] G. Patat and M. Sabt, "Please remember me: security analysis of U2F remember me implementations in the wild," in *Proceedings of 18ème Symposium sur la sécurité des technologies de l'information et des communications, SSTIC*, Rennes, France, June 2020.
- [38] H. Li, X. Pan, X. Wang, H. Feng, and C. Shi, "Authenticator rebinding attack of the UAF protocol on mobile devices," *Wireless Communications and Mobile Computing*, vol. 2020, no. 1, 14 pages, Article ID 8819790, 2020.
- [39] H. Feng et al., "A formal analysis of the FIDO UAF protocol," in *Proceedings of 28th Network And Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2021.
- [40] Yubico, "YubiKey," 2021, [Online]. Available: <https://www.yubico.com/products/>.

Research Article

An Efficient User-Centric Consent Management Design for Multiservices Platforms

Paul Marillonnet ^{1,2}, **Mikaël Ates** ¹, **Maryline Laurent** ² and **Nesrine Kaaniche** ²

¹*Entr'ouvert, Paris 75014, France*

²*SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, Évry 91000, Paris, France*

Correspondence should be addressed to Paul Marillonnet; pmarillonnet@entrouvert.com

Received 6 February 2021; Revised 22 April 2021; Accepted 10 June 2021; Published 22 June 2021

Academic Editor: Ahmad Samer Wazan

Copyright © 2021 Paul Marillonnet 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.

This paper presents an efficient user-centric consent management system to access online services of the Territorial Collectivities and Public Administration (TCPA) as well as user-authorized third parties. It defines a novel PII manager that supports a set of sources obeying to different authorization and PII retrieval protocols. This contribution is motivated by the necessity to interface TCPA services with remote sources that provide Personally Identifiable Information (PII). Hence, the originality of our solution is multifold. First, the burden for enforcing the interoperability between the sources and the TCPA services collecting the PII is reduced from the point of view of the user, the administrator of the User-Relationship Management (URM) platform, and the territorial agent responsible for processing the user's queries. Second, it defines a unified consent model supporting four types of sources. Third, it goes into details of practical implementations. Fourth, the relevance of the proposed PII manager for a relevant TCPA use case is demonstrated through a functional analysis.

1. Introduction

Our contribution pertains to the field of access control of Territorial Collectivities and Public Administration (TCPA) online services to their citizens. These TCPA are local and national official entities providing online services to citizens. Users of TCPA are requested to submit some regulated administrative requests, e.g., official document renewal, various allowance requests, and registrations to local services.

To benefit from these services, the user must provide Personally Identifiable Information (PII). As an example, the user issuing a passport renewal request is asked to fill in a web form including his PII fields, uploading scanned documents and retrieving PII from third-party sources.

Scanned documents and third-party-issued PII enable the user to provide the TCPA with validated data as valuable input for processing their requests. For instance, the France Connect official identity service provides user identity information, complying with the OIDC [1] identification

protocol. Implicit grant OAuth 2.0 [2] authorization is also at experimental stage for tax and children allowance information.

User-centric architectures that provide consent management aim at allowing the user to have the governance of their PII through their lifecycle, that is, the consent to collection, the usage control over time, and the visibility of past collections. These capabilities should happen even when the PII has been provided by third-party sources.

However, many shortcomings of user-centric PII management within TCPA have been identified over the previous years in Chapter 1.2.2 in [3]. The problem of the reunification of personal data has been well identified in the literature in Chapter IV in [4] and remains relevant. In fact, academic and industrial solutions, either they be (i) personal data stores [5–8], (ii) identity managers [9–13], (iii) anonymous certificate systems [3, 14, 15], or (iv) delegation architecture [16, 17], do not address the specific needs of PII management within TCPA.

Indeed, (i) personal data stores only support simple PII management scenarios where PII is collected on an “all-or-nothing” basis.

Similarly, (ii) identity managers by nature have limited support of third-party PII sources nor do they provide thorough delegation capabilities.

Furthermore, (iii) anonymous certificate systems, although a powerful solution for PII certification in a privacy-compliant manner, do not appear entirely suitable for PII management within TCPA, as too many core features of such management are out of scope of this category of solutions.

Finally, (iv) the same scope issue applies to delegation architectures when it comes to PII validation and to the support of third-party PII sources.

More precisely, none of these solutions addresses the four critical functional requirements identified for a standard TCPA’s use case, namely, (a) the need to manage the various consent information given by the user over time, (b) a wide extent of delegation on behalf of the user, (c) the possibility to validate PII, and (d) the support of remote PII sources.

In the European Union, functional requirements regarding user data management in TCPA have recently changed as part of the international regulations. As a matter of fact, the General Data Protection Regulation (GDPR) [18], applied by each member of the European Union since May 2018, requires that data controllers adopt new strategies regarding PII management of their users. Those strategies must address the need to give online users a thorough governance of their PII whether these PII be, for instance, service-provider transactional data or user profile data (additionally, some TCPA still partly rely on scanned documents). The necessity to deal with remote sources providing user PII led to propositions regarding identity-matching issues [19].

However, existing solutions did not discuss three other concerns that arise in the specific context of the TCPA.

First, user consent must be enforced consistently regardless of the PII’s actual location on any remote source. This paper aims at providing a way for the user to define consent to offline PII processing, wherever the PII. The fact that the PII is provided by remote sources does not hinder the consent management capabilities of the contribution.

Second, the interoperability concerns, that arise when dealing with such a heterogeneous system involving different sources, must be addressed.

Finally, the level of trust granted to remote sources for their role in providing user’s PII must be formalized, with the possibility for a PII provided by an untrusted source to be relevant for our TCPA use case, but less relevant than a PII originating from a fully-trusted source. This paper aims at specifying the levels of trust granted to sources and their impact on the TCPA use case.

This article describes a case-study architecture for TCPA services with the primary concern of enforcing users’ informational governance. The main proposition of this paper is a PII manager which supports the TCPA requirements with regard to PII management.

This paper is structured as follows:

Section 2 defines the use case of our contribution. Section 3 defines the system model, i.e., the actors, the functional requirements, as well as the environment and technical hypotheses of our contribution. Section 4 describes the related works, i.e., the academic or industrial solutions that are closely related to the use case. Section 5 introduces the PII manager within an existing architecture. Sections 6–8, respectively, present the PII Query Interface, the source backend, and the PII Management User Interface. Section 9 provides a functional analysis of the PII manager, proving its adequacy to the initial use case. Eventually, Section 10 presents a software proof of concept for our contribution and also provides implementation guidelines, before concluding in Section 11.

2. Use Case

This section proposes a use case for motivating our contribution and functional requirements. Our use case takes place in the Territorial Collectivities and Public Administration (TCPA) with which citizens interact. A number of PII are transferred through the TCPA and require a clear management.

In that context, our precise use case is about a user owning a proof of postal address, which is a PII, and delivering the proof to the TCPA for a procedure. In our contribution, a PII manager is responsible for managing the user’s data on his behalf when completing online procedures with their TCPA. Indeed, whenever needed for a procedure, e.g., the user’s child school registration or the user’s identity documents renewal, this data is directly made available by this service, only in case the user has previously agreed so by an explicit consent.

Their proof of address and other pieces of PII are managed for that service. The sources for this PII may be multiple, but the management features offered by this service remain unchanged.

This PII is used by various TCPA services possibly belonging to various collectivities at different scales such as town, department, region, and country. The user can visualize the past collections of their PII by the TCPA services and can revoke any further collection at any time.

3. System Model

This section defines the system model for our contribution. First, the actors of the use case are defined, along with their roles in the environment. Second, environment hypotheses are defined. Third, functional requirements, which our contribution must enforce, are also detailed. Last, the technical hypotheses of our industrial environment are listed.

3.1. Actors. The use case involves the four following actors:

- (i) The user of the online TCPA services submits one or several requests to the administrative or territorial

services. The requests are tracked through the user's account on the platform.

- (ii) The PII manager is responsible for enforcing the user consent and for evaluating the trust level of remote sources.
- (iii) The TCPA User-Relationship Management (URM) platform acts as a service provider for the user and relies on the PII manager service.
- (iv) The data sources may be official, i.e., maintained or acknowledged as such by TCPA, or private, i.e., maintained by a third-party service provider.

3.2. Environment Hypotheses. The PII managers are assumed to be dynamically discovered. This is made possible thanks to the user selecting the PII manager among a list for the TCPA platform to interconnect with.

The PII managers are also assumed to be regulated. This enables the TCPA URM platforms to establish direct trust with the PII managers, the latter having the critical duties to trustfully select PII sources and validate the retrieved PII.

Regulation can be enforced in two different hypothetical ways. First, there might be a regulation for a passlist of PII operators hosting several PII managers. The users would then be asked freely to choose the operator of their PII manager. Second, a PII manager authority, trusted by the TCPA, might organize PII managers based on a hierarchical certification architecture (i.e., a public-key infrastructure).

Finally, there might be an interest for the users to rely on several PII managers instead of only one. The direct advantage of distributing the responsibility for managing the PII over several entities would be higher availability of the service and distributed knowledge about their PII.

3.3. Functional Requirements. A noncomprehensive list of functional requirements relevant to PII management includes (i) user governance requirements, such as consent management, privacy/usability tradeoff, extent of delegation, and PII sharing capabilities and (ii) data exchange flow requirements, e.g., the supported PII types, the possibility to validate PII, the reusability of previously uploaded PII, and the support of remote PII sources.

As a result, the following requirements should be fulfilled by the PII manager.

- (1) Usage definition: the user can define the purposes justifying the PII collection.
- (2) Consent management: the PII manager keeps track of the authorizations given by the user for each piece of PII.
- (3) Usage monitoring: the user is given clear metrics of the PII consumption by any TCPA service. This monitoring facility offers a view of the user's PII usage by the TCPA services.
- (4) Delegation capabilities: the PII manager is able to decide whether or not to grant access to the PII, even when the user is not connected to the platform. In

that way, the access granting process is asynchronous from the authorizations granted by the user.

- (5) PII location abstraction: the PII manager ensures PII management regardless of the original source of the PII.
- (6) Protocol standardization: through standard interfaces, the PII manager can be queried with a common interface relying on standard PII management protocols.
- (7) Access uniformization: the multiple PII data sources are accessible in the same way.
- (8) Authorization protocol interoperability: the main identity management protocols, access mechanisms and authorization schemes, are supported with the heterogeneous remote sources to achieve interoperability.

3.4. Technical Hypotheses. The four following types of sources are considered:

- (1) Plain OAuth 2.0 resource servers based on OAuth 2.0 providers or OIDC providers
- (2) SAML 2.0 identity providers [20]
- (3) Plain read-only REST [21] sources accessible after an HTTP basic authentication [22]
- (4) Sources acting as resource servers according to the Kerberos [23] protocol

These sources have been chosen for their actual use in production environments. Additionally, they have not been designed to be interoperable, challenging the PII location abstraction requirement identified earlier in Section 3.3.

Interoperability concerns are addressed at the PII exchange protocol layer. The PII formats used in those protocols are (mainly) JSON for OAuth, OIDC, and REST and XML for SAML. We take the hypothesis that such nomenclatures are used for every given type of PII by all the sources. These nomenclatures are already in use in the TCPA environments. For instance, date and datetime information rely on ISO 8601 [24] and its use on the Internet, as covered by the RFC 3339 [25]. Similarly, the standardization of phone numbers as URIs is covered in RFC 3966 [26].

Additionally, this paper assumes that the different acting entities' clocks are loosely synchronized, which is common and considered easy to achieve.

Additionally, it assumes that the URM platforms do not permit the same identifier to be assigned to several users time after time. This is a loose requirement as most of the identifiers are either reversible or irreversible high-entropy pseudonyms, where reversible pseudonyms rely on symmetrical cryptographic functions, whereas irreversible pseudonyms rely on hash functions. In both cases, provided that the user PII being used as input to the pseudonym function varies, the risk of collision is considered negligible.

4. Related Work

This section presents the related work published in the literature. The literature provides several industrial or academic solutions, implemented or only provided with implementation guidelines. Table 1 gives an overview of existing solutions and provides a comprehensive comparison between them with respect to their support of various functional requirements and other identified technical considerations.

INDIGO [17] provides a token translation system that supports interoperability among sources supporting different protocols.

Its coverage of the functional requirements as well as the technical considerations is thorough; however, it does not give information about usage definition capabilities.

User-Managed Access (UMA) [16], specified by the Kantara Initiative consortium, provides the delegation capabilities of interest and benefits from protocol interoperability as specified in the OAuth 2.0 authorization protocol; for instance, see the OAuth 2.0 assertion framework [2].

The UMA use case does not cover authorization protocol interoperability nor does it provide an abstraction of the PII location (however, compatible with OAuth 2.0 token exchange [27]). PII usage definition is not strictly covered by this solution, as it depends on the actual implementation of an UMA architecture.

Additionally, the Kantara Initiative provides a consent receipt model [27] for generic applications where user consents need tracking. For specific applications where this model needs to be reduced, the Kantara Initiative also provides a Minimal Viable Consent Receipt (MVCR) [28].

Databox [7] supports PII sources and their respective drivers that provide an interesting architecture for our use case.

It does not support information about whether usage definition and protocol standardization are supported. Authorization protocol interoperability depends on the presence of drivers for these protocols.

The Fargo [8] document storage service has a rather limited functional coverage.

In spite of supporting interoperability, through a direct authenticated API or through OAuth 2.0, it shows many shortcomings either in the identified functional requirements or the technical considerations, such as the inability to handle raw PII or the absence of delegation capabilities. It is therefore not suited for our use case.

5. Our PII Manager as Part of the TCPA Architecture

5.1. Overview. This section details the way the PII manager acts within our architecture. The PII manager enforces the use case of Section 2, while maintaining the functional requirements of Section 3.3. The components of the PII manager are later described in subsequent sections, i.e., Sections 6–8.

5.2. Presentation of the Solution. The overall architecture is depicted in Figure 1. Figure 1(a) illustrates the global architecture involving our PII manager interfacing to the URM system and the remote sources.

Figure 1(b) depicts the interaction between the PII manager and the URM platform(s). This figure illustrates the need for a user identifier mapping service, presented in Section 7.3.2, as the user already has its own local identifier. For organizational reasons, the URM platforms maintain their own local identity manager. Indeed, each URM platform is maintained by a TCPA. These local identity managers act as local authorization servers within definite sectors. The term of “sector border” in this figure denotes the logical separation of identifiers between the URM platforms.

Figure 1(c) shows the interactions between the PII manager and the sources. The drivers, as part of the PII manager’s source backend presented in Section 7, make it possible to interface with several remote sources. This figure illustrates the use of a third-party authorization server (AS), as part of the OAuth authorization process for OAuth-based sources, labelled (a) on the figure, for getting the adequate access token for a given resource. (i) for instance, the France Connect data providers obeying to the OAuth 2.0 implicit authorization grant in Chapter 4.2 in [2]. Generic REST sources, labelled (b) on the figure, simply rely on base HTTP authentication mechanisms directly performed by the source, see for instance [22, 29]. (ii) for instance, the Particulier API (<https://particulier.api.gouv.fr/> (resource in French)). Alternatively, sources acting as Kerberos management resource servers, labelled (c) on the figure, refer to permission tickets that are granted in a two-step authorization procedure requiring first to get a ticket-granting ticket (TGT) from the key distribution center (KDC) and second to get a permission ticket from the ticket-granting server (TGS). (iii) it happens in the collectivities information system where the Kerberos protocol is used to access to network resources.

Eventually, Figure 1(d) depicts the user-centric PII management zone, through which the user manages their PII, the authorized sources, and their consent to URM platforms. It corresponds to the direct interactions between the user and the PII Management User Interface of our contribution, presented in Section 8.

To better explain the PII collection process, we also present the interactions between entities along with four sequence diagrams as follows:

- (i) A simple sequence diagram, depicted in Figure 2, describes the PII manager’s discovery by the TCPA URM platforms.
- (ii) Figure 3 describes the user authentication and consent obtention on the PII manager. Unauthorized PII access requests result in the obtention of a permission ticket. After user authentication and consent obtention, this ticket enables the issuance of an access token with the adequate authorization scopes on the requested resource(s).

TABLE 1: Related personal data management solutions comparison.

Criterion		Solution				This contribution PII manager
		INDIGO architecture [17]	UMA [16]	Databox architecture [7]	Fargo [8]	
Functional requirements	Usage definition	?	●	?	✗	●
	Consent monitoring	✓	✓	✓	✗	✓
	Usage monitoring	✓	✓	✓	✗	✓
	Delegation capabilities	✓	✓	✓	✗	✓
	PII location abstraction	✓	✗	✓	✗	✓
	Protocol standardization	✓	✓	?	✓	✓
	Access uniformization	✓	✓	✓	✗	✓
	Authorization protocol interoperability	✓	✗	●	✓	✓
Technical considerations	Identified consent model	?	✓	?	✗	✓
	Available implementations	✓	✓	✓	✓	✗
	Open specifications	✓	✓	✓	✓	✓

✓: yes, ✗: no, ●: depends on implementation, and ?: no information available.

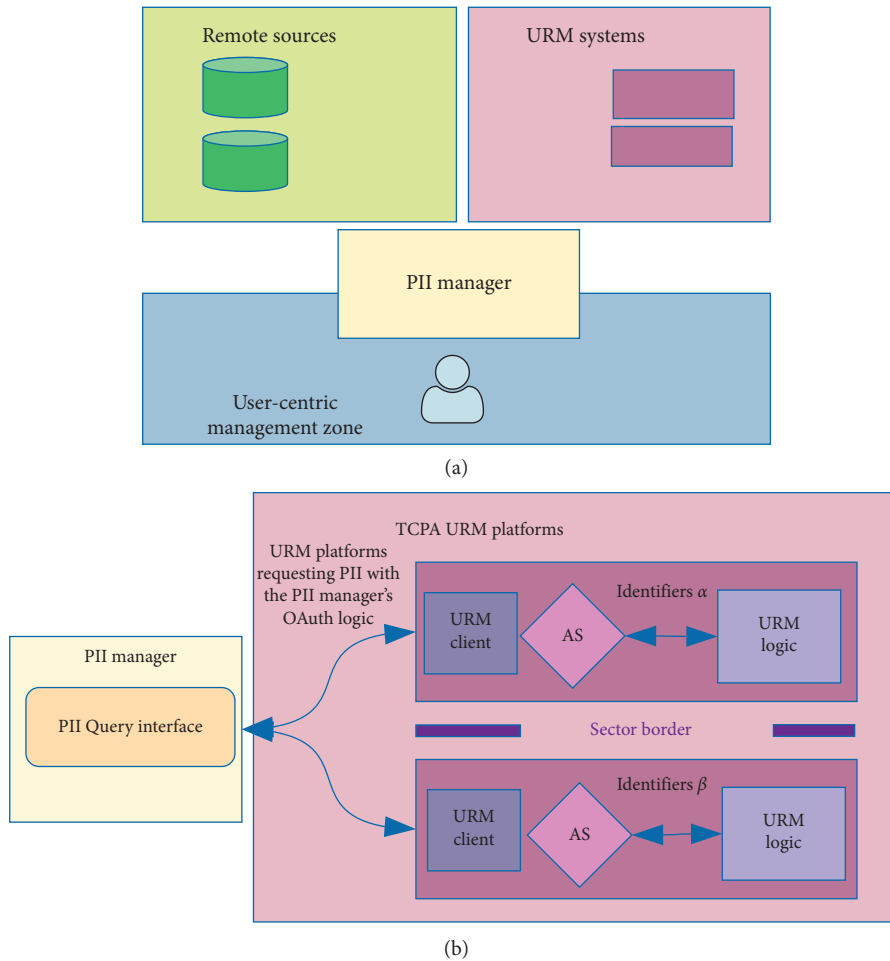


FIGURE 1: Continued.

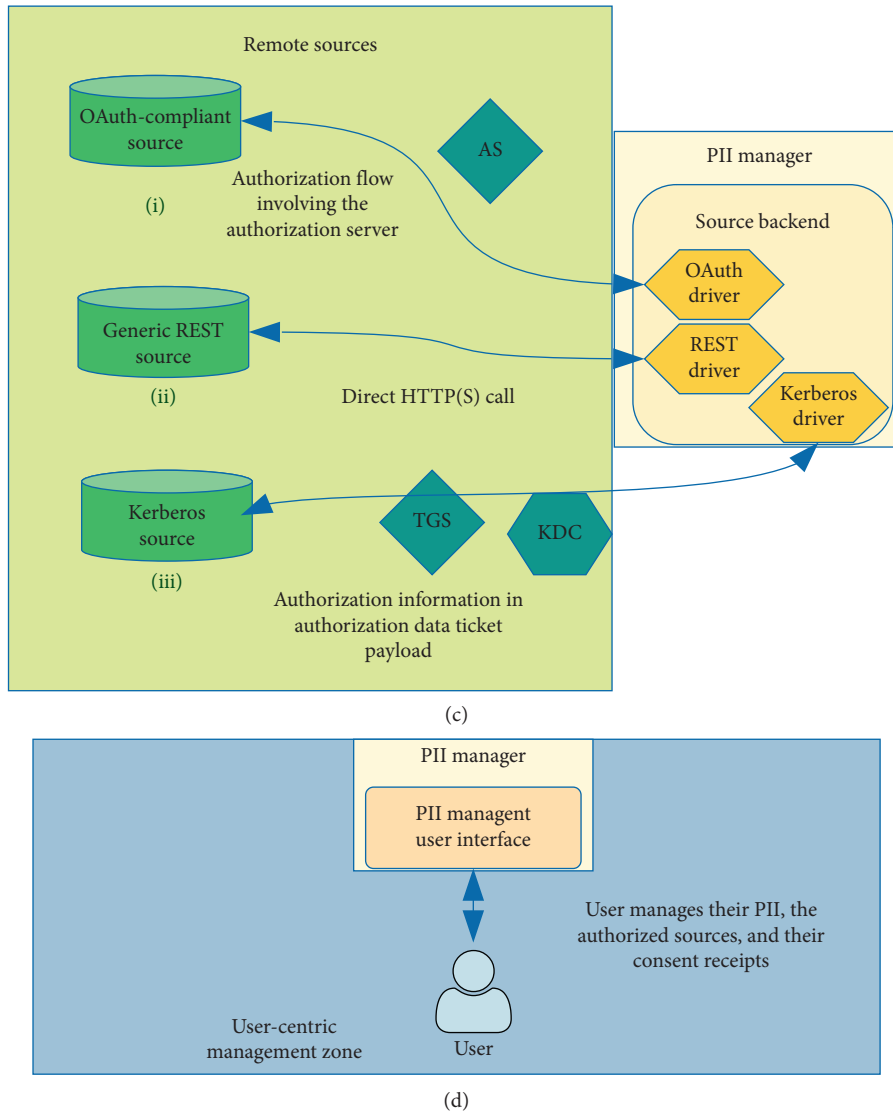


FIGURE 1: Complementarity in the PII manager’s roles regarding our use case entities. (a) General overview. (b) At URM platform side. (c) At remote sources side. (d) In the user-centric management zone.

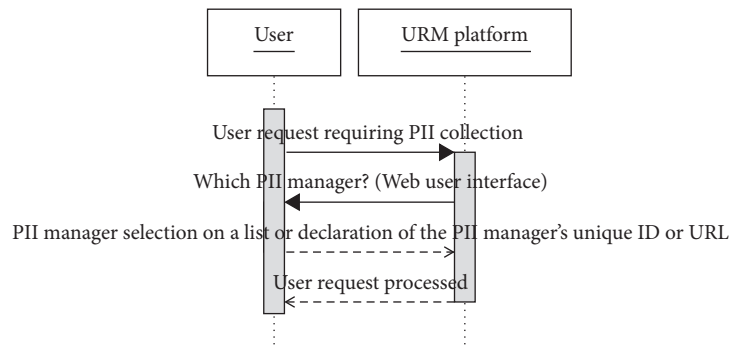


FIGURE 2: Discovery of the PII manager.

(iii) A sequence diagram for a typical PII collection scenario is provided in Figure 4. It shows the TCPA URM platform interacting directly with our PII manager, regardless of the data sources’ location. In

a two-step process, PII are collected by the TCPA URM platform. First, a request is sent by the TCPA URM platform to our PII manager through the PII retrieval endpoint. Second, the source backend at

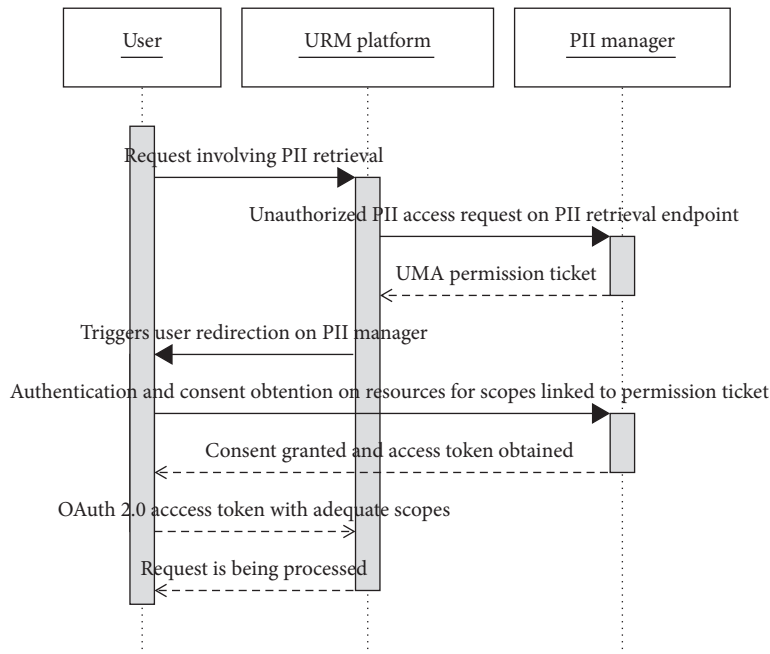


FIGURE 3: User authentication and consent obtention on the PII manager.

the PII manager selects the adequate driver for collecting the needed PII from the appropriate remote source. The authorization, which is a part of the second step, is either synchronous or asynchronous, unbeknownst to the requesting TCPA URM platform.

- (iv) Figure 5 describes the PII collection process once the source-side user authorization has been obtained. The PII manager stores his authorization information granted by the user. As long as this authorization information still has a valid time-to-live value and the user has not revoked the authorization, user interaction is no longer required.

The three different components of our PII manager, i.e., the PII Query Interface (PQI), the source backend (SB), and the PII Management User Interface (PMUI), and their mutual articulation are discussed in Sections 6–8.

First, Section 6 presents the PQI as a means to support PII location abstraction and consent management. Section 6 describes a set of endpoints, respectively, for registration, retrieval, and introspection of PII, the standard usage of this interface, and the consent management at this interface level.

Second, Section 7 introduces the SB which serves to enforce the authorization protocol interoperability. This section tackles the consent management, and it describes the support of OAuth token exchange.

Third, Section 8 describes the PMUI for supporting consent management and usage definition. User-definable parameters as part of this interface are also discussed in Section 8.2.

6. PII Query Interface (PQI)

6.1. Overview. The PII Query Interface is used by the TCPA URM services to retrieve the user’s PII on the PII manager. This section presenting the PQI is organized as follows: first, an overview of the PQI endpoints is given. Second, the registration endpoint and its ability to handle different types of registration information are discussed. Third, the usage of the PQI as part of our PII manager in the federated-identity environment of our use case is further described. Fourth, the role of the PQI in enforcing user consent at the PII manager’s level is also defined.

6.2. PQI Endpoints. The PQI is used by the URM platform when issuing requests to the PII manager. It exposes three endpoints:

- (i) Client registration as defined in Section 6.3.
- (ii) The PII retrieval endpoint complies with standard OAuth 2.0 scenarios for a resource server. Additionally, it performs the reduction of OAuth scopes, as mentioned in Section 3.3.4 in [16]. Eventually, the authorization process that is part of the PQI relies on the consent receipts generated by the PII manager.
- (iii) Access to the PII metadata introspection endpoint is legitimate to services that do not need the actual PII content. Metadata about this PII can be provided to them instead. Metadata include creation and modification informations and user consents granted to the requesting service for that PII.

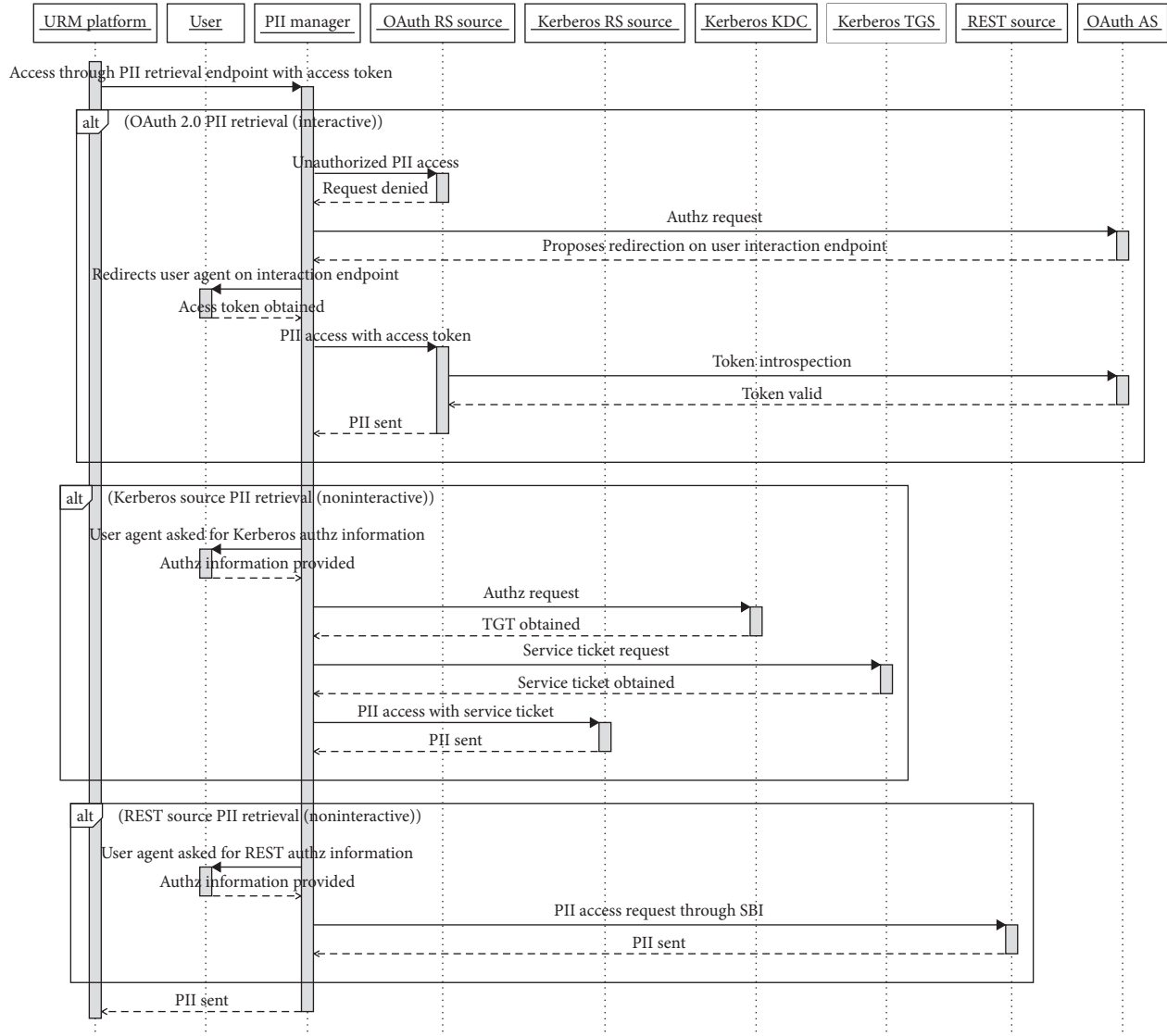


FIGURE 4: PII collection sequence diagram for getting the source-side user authorization.

In particular, user consent management is necessary to ensure that later offline authorization flows can be granted to the service.

6.3. *The Registration Endpoint.* The URM platforms must be registered beforehand.

The registration endpoint operates according to the specifications provided in [30]. To enable a service provider to access the registration endpoint, an access token must be delivered by the URM platform administrator. The issuance of the access token is a manual process and is not covered in this document.

Registering a platform is performed by a dedicated endpoint of the PQI, i.e., the registration endpoint. The following information needs to be provided while performing the registration:

(i) *Functional Registration Information.* This information includes terms of the policy, version of the

policy terms, categories of PII that will be collected, and purpose of the collection. It details all the elements that will be used when generating a consent receipt, if the user decides to give his consent.

(ii) *Technical Registration Information.* This information includes a set of redirection URIs. These URIs will be later used by the PII manager during the PII authorization and PII access process.

In return, the platform is given an identifier (“*client ID*”) and a password (“*client secret*”), necessary for all the future PII access requests. The platform is supposed to securely store these two registration elements, as they are required when issuing PII access requests to the PII manager.

When deploying the PII manager in our URM platform, each URM subservice (agenda, content-managed system, identity manager, Web form manager, etc.) is preregistered. This preregistration spares the user or agent from manually registering these trustworthy URM subservices. It is

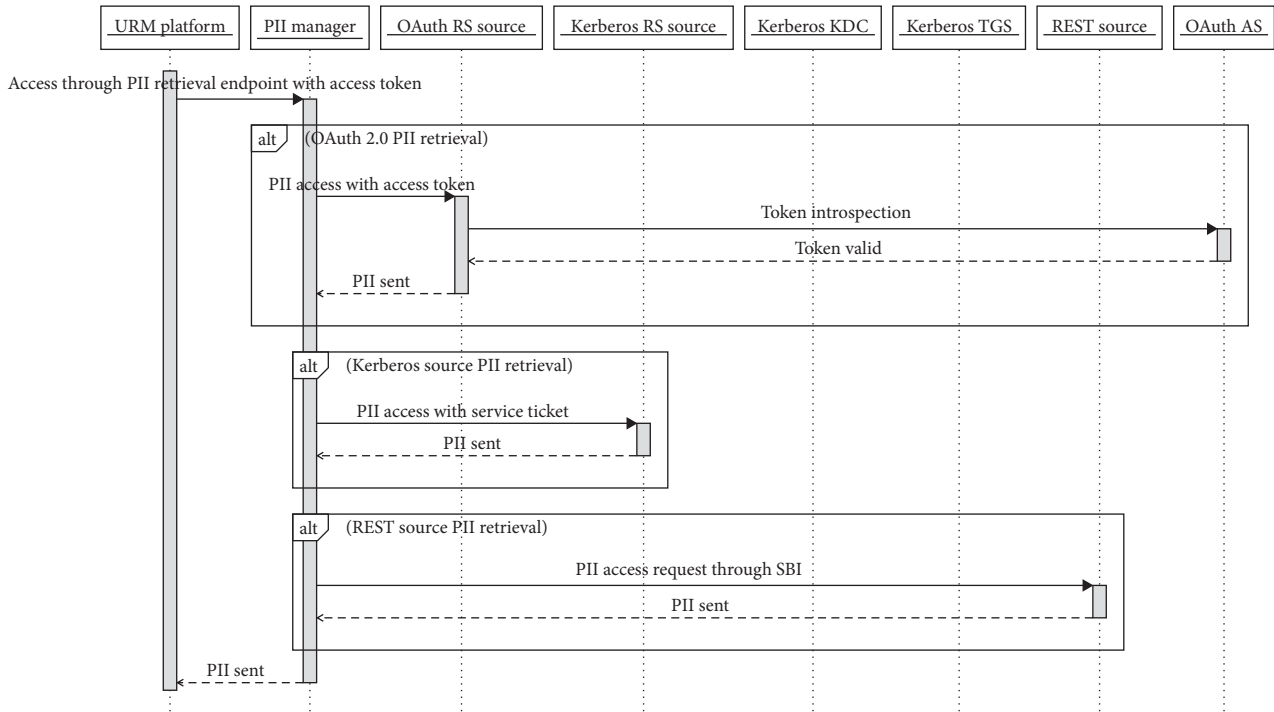


FIGURE 5: PII collection sequence diagram after source-side user authorization has been obtained.

performed by generating a long-lived registration API token that only these URM subservices know.

Alternatively, the URM platforms are issued a registration token to perform dynamic platform registration according to [30]. In return, these providers are able to access the registration endpoint as an API, issuing all the information necessary to register the platform.

6.4. Usage in a Federated-Identity Environment. The PQI for collecting PII from several remote sources is illustrated in Figure 6.

These three figures illustrate the different HTTPS redirect flows happening between the user and the PII manager when collecting PII from remote sources.

Figure 6(a) depicts the PII access process when the requesting URM client does not possess an access token. As shown in this figure, the PII access process provides an abstraction of the resource location.

Figures 6(b) and 6(c) describe the two possibilities for client-initiated PII collection, after a URM client access token has been issued, as depicted in Figure 6(a), i.e., (a) the direct PII collection for which the PII manager is doing the token verification on its own prior to send the PII to the URM client and (b) the indirect PII collection where the identity provider, through its token introspection endpoint, is asked to verify the token.

6.5. User Consent Enforcement at PQI Level. User consent enforcement at the PQI level enables (i) the delegation of access decisions on the PII manager and (ii) the generation of consent receipts for traceability purposes. Properties (i)

and (ii) have both undergone specification efforts by the Kantara Initiative.

An access decision delegation, based on OAuth 2.0 scope-based access requests, is shown in Section 3.3.4 in [16]. UMA specifies (i) the delegation of access decisions. The relevant algorithm, presented in Section 3.3.4 in [16], deals with the way the UMA server should decide on whether to grant access to users. Given an OAuth client *C*, the input information for this algorithm to be run is preregistration scopes for *C*, recently requested scopes for *C*, and OAuth scopes associated with the resources requested by *C*.

With this input information, the authorization server evaluates which scopes of authorization can be granted to the client. Therefore, three cases are possible: the authorization server to either grant the authorization with the actual scopes as requested by the client, to restrict the authorization grant to a smaller set of scopes, or to completely deny the authorization request.

This algorithm ensuring properties (i) and (ii) is detailed in Section 8.2.

7. The Source Backend

7.1. Overview. The source backend makes it possible to deal with multiple sources, with a backend for each source type. It enables interoperability through the support of multiple authorization and PII access protocols. Indeed, the multiplicity of such authorization protocols makes it hard to identify a unified consent model.

An OAuth scope is an authorization unit, characterizing a resource or an action to be performed on it. In OAuth terms, it is a character string of blank space-separated

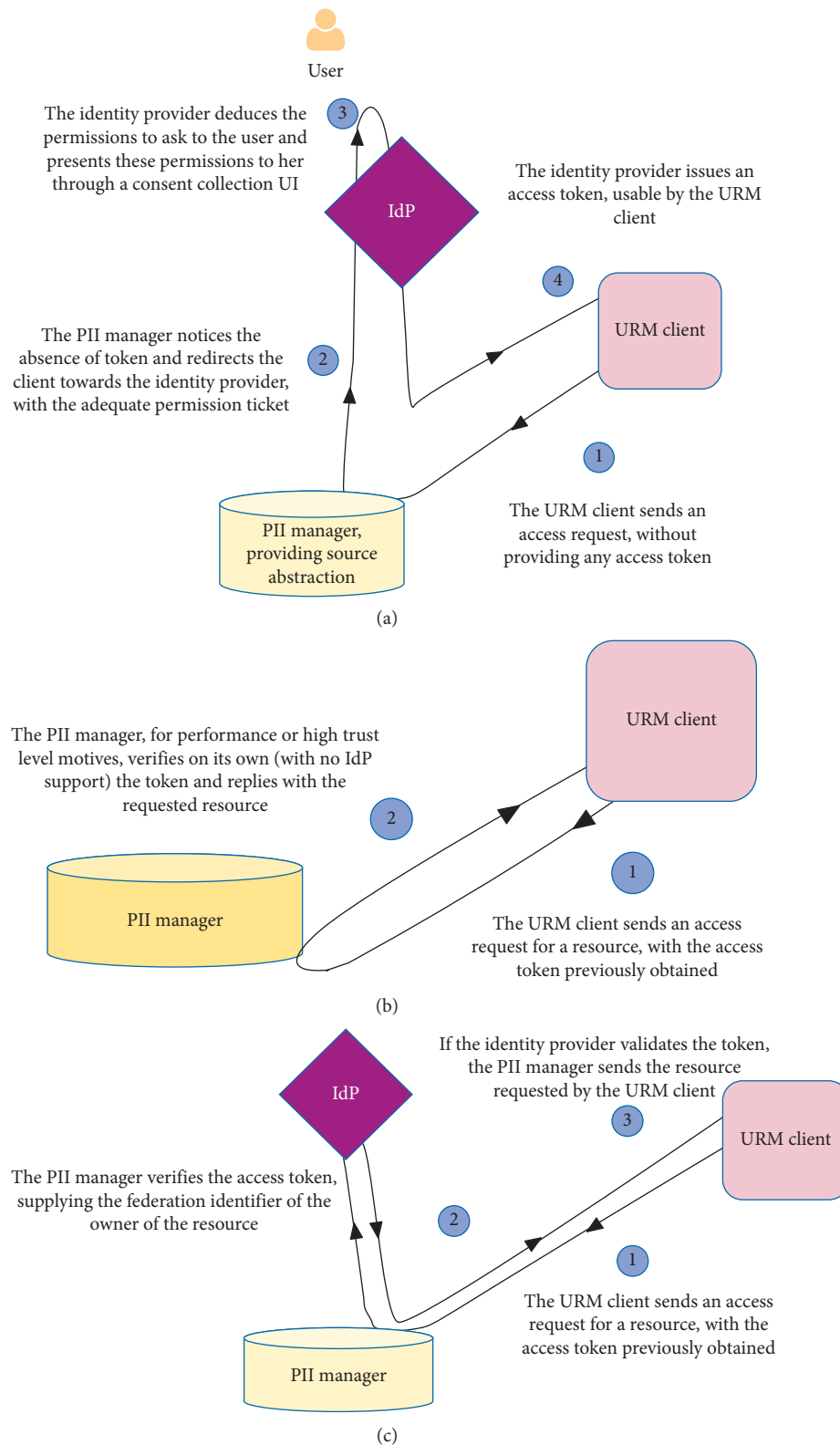


FIGURE 6: PII collection by the URM client. (a) Access token issuance by the user. (b) Direct user token consumption. (c) User token consumption after verification by the IDP.

keywords that constitute the scopes set. The set of scopes is used in the PQI retrieval endpoints to specify the PII requested.

Finally, this section also discusses consent management as part of the source backend when interfacing with remote sources. For each type of sources, a study of the

authorization information structure is given. Strategies to map authorization information to our consent model are defined.

7.2. Supported Protocols. The list of supported protocol is as follows:

- (i) An **OAuth 2.0 (including OIDC) provider** does not require any translation, as it is directly usable by the UMA authorization process used within the URM platform. The scopes used by the OIDC identification protocol are directly compatible with UMA.
- (ii) A **SAML provider** translation mainly relies on the use of the OAuth 2.0 assertion framework [31] and its use with the translation of SAML 2.0 assertions [32]. These specifications provide “out-of-the-box” processes for translating SAML assertions to OAuth 2.0 authorization information. Thus, SAML assertions can be used either as OAuth 2.0 client authentication information or authorization grants.
- (iii) The **Plain read-only REST sources** only require a static set of scopes predefined by the URM platform administrator. A direct mapping between HTTP verbs as used by REST sources and standard OAuth scopes used by UMA can be established.
- (iv) The resource server operating according to the **Kerberos** authorization protocol needs a valid permission ticket. The ticket scope always concerns access to a resource. Finer scopes of authorizations are not supported by the protocol (Kerberos tickets contain an optional authorization data field that can be used to implement authorization scope support. However, it is not covered by the specification, see Chapter 5.3 in [23]. The way the PII manager manages the Kerberos session key, and the manager’s registration in the principals’ database maintained by the Kerberos administration server, is out of scope of this paper as it deals with the registration process more than the authorization.

7.3. Consent Management

7.3.1. General Considerations. The main objective of consent management is the respect of the user’s choices when it comes to considering the URM platforms’ PII queries. The authorization system of the PII manager hence relies on such consent management. The authorization lifecycle is therefore managed by the user.

In particular, consent management on the architecture implies keeping track of the users’ previous choices regarding the collection of their PII by service providers. The user’s consents have a limited lifetime, in particular, they can be given for a single immediate collection, and they have scope denoting the extent of the granted authorization.

We can still define the basic capabilities of the consent management part of the proposed solution. These capabilities are derived from the consent receipt model defined in

[27]. This consent receipt model formalizes user authorizations in the consent management system.

The information contained in the receipts as defined in [27] belongs to three categories: (i) receipt transaction fields, (ii) transaction parties’ fields, and (iii) data, collection, and use fields.

All three categories are relevant to enforce consent management as per our consent model. In particular, transaction parties’ fields enable the declaration of PII controllers that collect the data and whether this collection happens on behalf of another controller. Additionally, data, collection, and use fields enable the declaration of termination policies that apply for the consent as well as purposes and PII categories that apply for the receipt.

A comparison table of the consent models supported by the PII manager is shown in Table 2.

The following criteria are used to make the comparison:

- (i) Revocability means the ability to cancel a previously given consent information.
- (ii) Multidomain management is relevant when, as depicted in Figure 1(b), several URM platforms interface with a single PII manager. The management of consent information across domains is therefore a key element.
- (iii) Terms-of-usage versioning refers to the ability to record the version number of the terms-of-usage for the PII collection that has obtained the user’s consent.
- (iv) Authorization scope means being able to specify unitary elements defining the authorization request.
- (v) Interuser resource sharing defines whether the consent information specifies the ability for other users to directly access the PII.
- (vi) Direct verifiability means whether the verifiability information requires a request to an authorization server or can be directly verified by the resource server.

The remaining of this section is organized as follows.

First, the user identifier mapping as part of the consent management is presented. Second, the translation of the authorization information according to the four protocols supported by the sources is detailed. For each protocol, a brief description of the authorization information structure is given, and a translation procedure is then introduced.

7.3.2. User Identifier Mapping. The PII manager is responsible for implementing the user identifier mapping across the sources and the URM platforms. It is of utmost importance that the collected PII from different sources for one user do actually belong to that user. In order to enforce this user uniqueness, the PII may rely on several possibilities, be it either a unique official identity sources such as France Connect, or automated mapping as presented in [19].

The TCPA deploys an identity provider that respects the identity-federation principles. In particular, it provides sector identifiers according to several service sectors of the

TABLE 2: A comprehensive comparison between different consent models.

Criterion	Consent model			
	PII manager consent receipt (Kantara)	OAuth 2.0 access token	Kerberos ticket	Standard ACLs
Terms-of-usage versioning	Yes	No	No	No
Direct verifiability	Yes, by definition of locally managed consent	Yes	No	Depends on model
Authorization scope	Yes	Yes	No	Partially supported
Revocability	Yes, by definition of locally managed consent	Yes, through AS	Yes	Yes
Multidomain	Yes	Yes	No	Depends on model
Interuser resource sharing	No	No	Yes	Depends on model

TCPA. As a result, two services belonging to different sectors are provided with two different identifiers corresponding to the same user. The PII manager is thus responsible for managing the user identifier mapping across the services.

First, the authorization information from a first URM platform, say platform α , may present the user information including an identifier u_α . A second platform β only knows the authorization information that contains a user identifier u_β . If the user's consent is applied on platforms α and β , a user-identifier mapping must be performed. The PII manager needs to provide a mapping function m :

$$m: \begin{array}{l} U_\alpha^* \longrightarrow U_\beta^*, \\ u_\alpha \mapsto u_\beta, \end{array} \quad (1)$$

where U_α^* and U_β^* are, respectively, the identifiers' sets of platforms α and β . Since those platforms support the OAuth authorization framework, these identifiers can be (i) pseudonyms, (ii) Universally Unique Identifiers (UUIDs), or (iii) human-friendly attributes such as the user's e-mail addresses. Regardless of the platforms' actual identifier policy, these identifiers are considered to be string characters.

Additionally, the PII manager acting as a resource server interacts with several authorization servers that do not belong to the same federated-identity environment. They therefore do not share the same user identifiers.

This altogether justifies the need for an identifier mapping endpoint.

The identifier is mapped to a pseudonym derived from (a) the source subject identifier and (b) the target sector identifier, in a similar fashion as presented in [1]. The process used to derive these pseudonyms can rely on nonreversible pseudonyms' identifier generation as presented in Chapter 8.1 in [1].

The target service then receives a pseudonym with a set of human-readable information that serves as input for the identity-matching procedure, as presented in Chapter 5 in [19].

7.3.3. Translation to OAuth (including OIDC) Consent Information

(i) *Structure of Authorization Information.* Access tokens, in their most common JWT [33] form, are structured as follows

(the structure may vary when the JSON token is encrypted and when it contains unprotected header):

- (i) A header bearing token metadata
- (ii) A (cleartext or encrypted) payload, which contains the core authorization information
- (iii) Optionally, a signature whose validation information is contained in the header

(ii) *Grant Type Translation.* Grant type translation happens when using standard-OAuth input authorization information within the (UMA-OAuth) PII manager. The OAuth-supported grant types are the implicit grant and the authorization code grant. The authorization code grant is a two-step grant allowing the URM client to request access tokens, by letting the authorization server know that it was giving the user's authorization. On the contrary, the implicit grant is simpler as no authorization code is involved. The user's consent given to the URM client directly results in the authorization server's response that includes an access token.

(iii) *Scope Translation.* Translating scopes is performed in three steps:

- (1) Identify the scopes of interest for the URM platform
- (2) List all the other scopes that can be part of the translated assertion
- (3) For each of these other scopes, provide a mapping to the scopes supported by the URM platform

(iv) *The OIDC Profile.* The only supported grant type is the authorization code. OIDC is a simplified version of OAuth, in which the identity provider is also the resource server (in particular, the identity information is the requested resource).

(v) *Direct Mapping.* The direct mapping from the PII manager's consent model to OAuth authorization information happens as follows:

- (1) Issuer to iss
- (2) Start timestamp to iat
- (3) Expiry timestamp to exp
- (4) Authorization resource to the resource URI of the authorization process

- (5) Authorization scope to the scopes
- (6) Authorization user identifier to sub

7.3.4. Translation to Kerberos Consent Information

(i) *Direct Mapping.* The direct mapping from the PII manager's consent model to Kerberos' permission ticket information happens as follows:

- (i) Issuer to cname and crealm
- (ii) Start timestamp to starttime
- (iii) Expiry timestamp to endtime
- (iv) Authorization resource to authorization data payload of the ticket
- (v) Authorization user identifier to principal

7.3.5. Translation to Plain ACL Consent Information

(i) *Structure of Authorization Information.* According to our hypotheses, plain ACLs' consent information is made of (i) authorization metadata (e.g., temporal and spatial validity metadata) and (ii) core authorization information (e.g., subject and object of authorization).

(ii) *Direct Mapping.* The PII manager's consent model maps to the ACL information. For instance, the following mapping applies:

- (1) Issuer maps to client
- (2) Start timestamp maps to beginson
- (3) Expiry timestamp maps to endson
- (4) Authorization resource maps to resources-uris
- (5) Authorization user identifier maps to subject
- (6) Delegation flag maps to delegated

7.3.6. Translation to SAML Assertions

(i) *Structure of Authorization Information.* The PII fields of a SAML assertion are as follows:

- (1) The subject of the authorization information (Subject):
 - (i) A technical identifier (NameID)
 - (ii) A set of human-friendly PII about the subject
 - (iii) Validation metadata
- (2) The authorization:
 - (i) The assertion statement
 - (ii) Additional assertion validation metadata

When the assertion is signed and/or encrypted, the public keys and algorithm declarations are available in the server's and service provider's respective metadata [34].

(ii) *Direct Mapping.* The SAML assertion translation specification of [32], as part of the OAuth 2.0 assertion framework presented in [31], is used for the SAML driver. Campbell et al. [32] specified the way SAML assertions can be used as authorization grants or as client authentication information.

Using this framework means that mappings for the elements of a SAML assertion are supported. For instance, the following elements need mapping:

- (i) The user identifier needs to be mapped to the UUID within the URM platform. This mapping is handled by the identity provider of the URM platform, which offers a user-identifier resolution service to the PII hub.
- (ii) Scopes of authorization need to be translated to the scopes that actually are enforced by the URM platform. There is no possible comprehensive list for these scopes, as the OAuth-based protocols can extend the standard scope model.

In terms of mapping the content of the SAML assertion to the consent model, the following elements need to be considered:

- (i) User identifier maps to the NameID. The NameID format must be one of UUID or e-mail address.
- (ii) Start timestamp maps to NotBefore.
- (iii) Expiry timestamp maps to NotOnOrAfter.
- (iv) Names, formats, and values of standard attributes also need mapping. This mapping depends on the type and format of attribute and is not covered in this document.
- (v) Our consent model also supports extended, admin-definable, attributes that can be mapped in a similar fashion, depending on their respective types and formats.

7.4. *Considerations Regarding Token Exchange.* As specified in [35], plain OAuth access tokens can also be exchanged for delegation or impersonation purposes. The delegation and impersonation (impersonation is not used in a negative way in [27]. It means that the target service does not need to be aware of the delegation that happens between a subject entity and an actor entity) features require that the PII manager be able to perform an additional round of redirection, that enables the retrieval of a security token.

In particular, it requires the ability to

- (i) Receive an access token and to choose the adequate backend
- (ii) Retrieve the newly-issued security token
- (iii) Submit the token for consumption to the proper backend

However, the Token Exchange IETF Request for Comments (RFC) is quite recent, and the use of such protocol in the industry is not widespread yet.

8. PII Management User Interface (PMUI)

8.1. Overview. The PII management interface's main purpose is to provide users with an interface for managing authorizations of URM Clients on their PII. The configuration capabilities of the component enforce such a management even when the user is offline.

User-Managed Access in Section 3.3.4 in [16] proposes a procedure to ensure these offline properties. The procedure relies on the OAuth scopes on resources requested by the URM clients.

Additionally, the PMUI reflects the ability of the PII manager to abstract the PII location. As a reminder, we note that the PII abstraction property requires that the PII manager acts first (i) as a requesting party for the registered PII sources and then conversely (ii) as a resource server for the URM platform.

This management user interface offers the visualization of PII transfers with service providers. When granting PII access to the SPs, the PII manager provides logs' information to the associated users and to the data owners. These logs can be visualized at the convenience of the user, i.e., whenever it is suitable for the user. Information obtained at client registration time is also presented to the user, such as the category of service providers and the purpose of collection.

Eventually, the user must be able to revoke a previously granted access. The PII Management User Interface thus includes management pages for each previously created access rule.

8.2. User-Definable Parameters. We rely on an access policy definition at resource registration time, enabling the user to define the parameters below:

- (i) The required scopes for the resource: these scopes describe usual operations such as reading, deleting, modifying a resource, and accessing a subresource. Alternatively, they can also be specific third-party application scopes.
- (ii) The time window of access authorization: when issuing access tokens within the URM system, the PII manager uses these user-defined parameters to adjust the validity time window of the token.
- (iii) The service or the category of service for this authorization rule: the services accessing to the user's PII are sorted according to user-defined categories. Any authorization rule defined by the user is applicable to a category of services only.

When user data are provided by sources acting as SAML or OIDC providers or as OAuth resource servers, this information may already be provided along with the PII payload. Yet, it may be overridden by the user. For plain REST sources, however, this information needs to be provided at registration time.

As a result, the user interface adopts this approach, letting the user define the aforementioned parameters when the metadata provided by the source does not provide this information.

The constraint of unicity regarding the resource, the scope of action, the time window, and the category of services altogether is enforced. At a particular moment in time, for a given resource, a category of service, and a scope of action, at most one authorization rule can apply. The access control system denies all by default.

The algorithm applying, based on the consent model given in [27], can be formulated as the verification:

- (i) Of the issuance date
- (ii) Of the expiry date
- (iii) That the terms-of-use version applies
- (iv) That the consent geographical location applies
- (v) Of the scopes according to previously-granted scopes for the category of service as follows:
 - (1) Retrieve the set of previously granted scopes as defined in Section 3.3.4 in [16]
 - (2) Translation of authorization information into OAuth scopes known to the PII manager
 - (3) If the authorization server chooses to reduce the set of granted scopes, in comparison with the requested scopes, then a reverse translation is necessary: the OAuth scopes known to the PII manager are reverse-translated to the OAuth authorization information model as dealt by the requesting party.

For instance, while accessing a REST source, the delegated authorization flow is the following one, as summarized in Figure 7:

- (1) URM client request: the URM (OAuth) client asks for access to resources/picture1.jpg with the read write print caption scopes
- (2) First way translation: the client requested scopes translate to the ability to perform GET, POST, or PATCH on the URI
- (3) Reduction: the PII manager gets from its own internal authorization information that only the GET verb is authorized for that resource and this URM client
- (4) Reverse translation: the PII manager reverse-translates to the read OAuth scope
- (5) PII manager response: the PII manager sends an access token with the reduced read scope

9. Functional Analysis of the Architecture

9.1. Overview. This section provides an informal analysis of the compliance of our PII manager approach with the targeted functional requirements described in Section 3.3. It starts first by listing below useful elements of our solution for fulfilling that compliance and then it provides a full description for some, in the sections that follow.

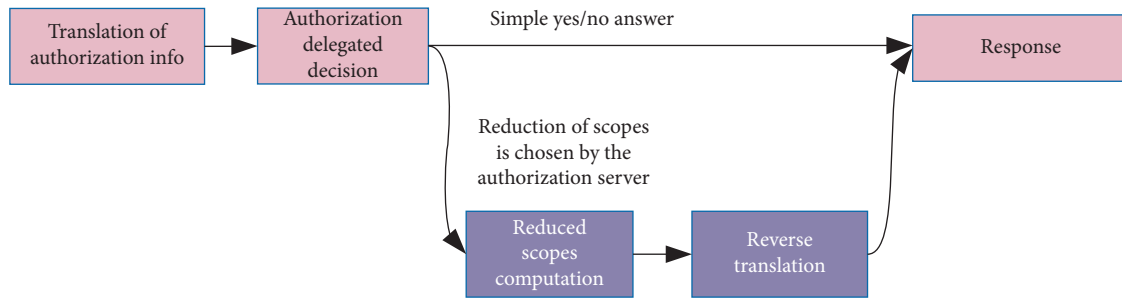


FIGURE 7: PII collection by the URM client: extension of the UMA decision process by the PII manager.

- (1) **The PII management capabilities** map to requirement “usage definition” (requirement #1)
- (2) **The PQI and source backend of the PII manager** map to requirements “consent monitoring” and “usage monitoring” (#2 and #3)
- (3) **The delegation capabilities** map to requirement “delegation capabilities” (#4)
- (4) **The PQI and the source backend**, again, map to requirement “PII location abstraction” (#5)
- (5) **The unified authorization scheme** maps to requirements “protocol standardization” and “access uniformization” (#6 and #7)
- (6) **The support of several types of sources** maps to requirement “authorization protocol interoperability” (#8)

9.2. Usage Definition (Requirement #1). The consent receipt model adopted in our contribution covers usage definition. Indeed, the data fields and the transaction fields that are part of this model make it possible to specify the purpose of PII collection as part of user consent information.

9.3. Consent Management and Usage Monitoring (Requirements #2 and #3). Consent management is achieved thanks to the use of consent receipts and the mapping of authorization information. Usage monitoring is ensured by PII manager, acting as a single resource server for the URM platform(s).

9.4. Delegation Capabilities (Requirement #4). Section 8.2 specifies the necessary delegation capabilities that our PII manager supports in order to comply with the use case. In particular, that section provides a pseudoalgorithm for the reduction of the authorization scopes set, compatible with the UMA-delegated authorization process.

9.5. PII Location Abstraction (Requirement #5). The translation of authorization information defined in Section 6 enables the PII manager to provide the TCPA URM platforms with the user’s PII regardless of the PII actual location.

9.6. Protocol Standardization and Access Uniformization (Requirements #6 and #7). The access control rules describe whether the user authorization information for accessing PII can be granted to a URM client, either directly or through inference based on contextual information.

The direct grant is performed through the definition of preferences by the user. Moreover, these preferences are directly linked to a service provider’s client.

This model helps ensuring the minimization of PII transfers: the PII manager, when deciding whether to authorize PII access to a given URM client regarding several categories of PII, can quickly verify if one of the categories of PII is not accessible by that URM client.

A simple PII verification algorithm is as follows:

- (i) Claims and scopes are checked against the rules defined for this URM client. The two main elements involved in this process are as follows:
 - (1) The translation of the source’s authorization information into scopes that are known to the authorization server. This translation step depends on the type of source, as explained in Section 7.
 - (2) The comparison of the required scopes with the user-defined preferences.
- (ii) When user-defined preferences are insufficient in order to take action, the required scopes are also compared to the scopes previously granted to the URM client. Based on the UMA OAuth 2.0 grant access control process provided in [16], the server can decide to reduce the required scopes to a set of scopes that are appropriate regarding the URM client and the requested resource. Alternatively, the resource server may reject the URM client’s request.

9.7. Authorization Protocol Interoperability (Requirement #8). As described in Section 5, respecting our use case involves a strong correlation between the PII management entity and the TCPA URM platform. We now discuss the (a) interoperability property and (b) more specifically the possibility for the PII management entity to interface with other TCPA’s URM platforms.

In order to ensure this interoperability property, four necessary subproperties are identified: (i) interface standardization, (ii) dynamic registration (or not configuration

at all), (iii) authorization protocol(s) standardization, and (iv) data exchange format(s) standardization.

- (i) It means that the PII manager can be used for several URM platforms at a time. This subproperty is ensured by offering a standard REST API. Such an API offers unambiguous data location format, standardized data operation syntax using HTTP verbs, and use of common Web technologies.
- (ii) It is necessary if that interoperability property is expected to be seamless, i.e., with no configuration whatsoever by any human agent involved. This is achieved by OAuth 2.0 Dynamic Client Registration [30] and its associated management protocol [36]. In order for the PII manager to perform dynamic registration of the URM platforms, the following information is necessary:
 - (a) Endpoints information (support grant types and token authentication methods)
 - (b) Redirection URIs
 - (c) Keysets' locations

From the user's point of view, when registering a new PII manager to their URM platform, it is sufficient to simply provide the PII manager URL. In delegated authorization mechanisms such as the UMA grant for the OAuth 2.0 authorization protocol, a URM tool acting as a Requesting Party needs to identify itself (with a prior registration on the authorization server) before obtaining the requested authorization data.

- (iii) It is provided when the PII manager acts as an OAuth 2.0 Resource Server. This PII manager is therefore able to verify the validity of an access token for a given Requesting Party. This validation is performed according to the authorization server's token introspection endpoint [37].
- (iv) It implies that the PII exchanged is presented in a way that is recognised by both the sender and the receiver. This standardization is enforced by the common use of OAuth-based protocols and the assertion framework that makes it possible to interface with other federated-identity management protocols such as SAML.

10. Implementation Considerations and Proof of Concept

This section describes the implementation considerations resulting from the prototype design of the PII manager in a TCPA URM software system.

10.1. Proof-of-Concept Implementation. A proof-of-concept implementation is visible at the following public git repository: (<https://git.entrouvert.org/pii-manager-poc.git/>). It is distributed under the Affero General Public License, in its third version (AGPLv3) (for more information about the AGPL and its differences with its more famous sibling the

General Public License (GPL), see (<https://www.gnu.org/licenses/agpl-3.0.html>). It provides a proof of concept for the support of OAuth 2.0 sources and REST sources.

It uses the Django web framework, in its second version. The noteworthy parts of the implementation are as follows:

- (1) The data model implements, amongst other models, the consent receipt model specified in [27].
- (2) The view logic performs OAuth authorization with the scope reduction logic presented in Section 8.2. The view logic implementation conceals the complexity of gathering PII from several sources from the user's point of view.
- (3) The source backend implements support for OAuth 2.0 sources and REST sources and puts the base layout for a future support of SAML and Kerberos sources.

Of course, some parts need improvement, such as the PMUI, presented in Section 8, which for now only relies on the Django administration user interface (“/admin/”) for the management of Django data models.

10.2. Considerations and Guidelines Regarding the Implementation of the PII Manager

10.2.1. Client Type. The OAuth client type depends on the ability of the client to store the secret information that was delivered by the authorization server. For instance, public clients are unable to safely keep a client secret and are therefore excluded from some authorization grant types. It is likely that the PII manager as an OAuth 2.0 client will be able to store its client secret and to operate according to any of the four main authorization grant types defined by the OAuth protocol.

10.2.2. Access Token Lifetime and Refresh Token Persistence. Providing PII abstraction can result in longer PII retrieval time by the PII manager. The lifetime of access token delivered by the PII manager for usage within the URM platform should take this extra delay into consideration, at the implementation level.

The choice to persist refresh tokens must be evaluated according to its privacy-usability tradeoff: persistent refresh tokens are more convenient for the PII Manager. On the contrary, for obvious reasons they make it more difficult to enforce client revocation.

More generally, adequately choosing token lifetime as well as authorization code expiration timestamp can help enforce privacy-compliant properties such as forward secrecy: a malicious user obtaining a token will be limited by its lifetime (that is, in the case of a plain bearer token [39], when cryptographic tokens such as JSON Web Tokens (JWT) are used, man-in-middle attacks such as the theft of a token are preventable).

10.2.3. Introspection Endpoint and Token Validation. The choice of performing token introspection by the

authorization server from which the token originates [37] must also be examined carefully while performing the implementation of the PII manager. In some cases, token introspection by the origin authorization server may not even be possible, in which case a partial validation would be the only possible option.

11. Conclusion

Our PII manager, presented into practical implementation level of details, provides an abstraction solving issues due to multiple sources being considered. These issues include variety of protocols being implemented by the sources, and the resulting variations in the authorization information and in the user consent enforcement. Our approach relies on three main specified components allowing the support of functional requirements identified in our use case, including the support of several sources.

The PII Query Interface (PQI) specifies the way the PII manager interacts with URM platforms, possibly involving an identifier mapping service.

The source backend (SB) specifies the interface with sources obeying to different authorization and PII retrieval protocols. Operating a SB requires a unified consent model, involving an authorization information translation across protocols. This consent model unification step, as performed by the SB, also needs a reverse translation step when the authorization scopes need reduction.

The PII Management User Interface (PMUI) specifies the user-definable parameters that take part in the support of multiple sources and the enforcement of user consent on the PII retrieved across these sources.

Our functional analysis of the architecture demonstrates that the requirements identified in the use case have been correctly addressed.

However, an inherent limitation due to the use case and its requirements can be identified. Indeed, this use case and requirements are based on the industrial hypotheses that have their own bottlenecks and limitations. First, the implementations' variations of effective sources from the theoretical specifications bring complexity to the solution and make it not entirely generalizable. For instance, the inner authorization logic within vanilla OAuth authorization servers (i.e., non-UMA servers) is out of scope of [2]. Having a clear inner authorization logic known to the PII manager when interfacing with OAuth sources would be beneficial, yet not possible at the moment. More generally, we acknowledge that, in our architecture, the PII manager, albeit necessary to enforce our use case while maintaining its functional requirements, results in adding an extra indirection layer that complexifies the overall architecture, induces extra costs, and requires more computing resources. Eventually, the industrial reality that new standards and protocol does not, for practical reasons, always mean the

actual depreciation of the previous ones, brings interoperability concerns, as it complexifies the implementation of the PII manager by increasing the number of required drivers.

Finally, solutions such as the PII format proposed by the System for Cross-domain Identity Management (SCIM) [38], although not supported by effective production official sources yet, would be a solution to that issue.

Abbreviations

API:	Application programming interface
AS:	Authorization server
Authn:	Authentication
GDPR:	General Data Protection Regulation
IDP:	Identity provider
KDC:	Key distribution center
PII:	Personally Identifiable Information
PQI:	PII Query Interface
PMUI:	PII Management User Interface
SB:	Source backend
TCPA:	Territorial Collectivities and Public Administration
TGS:	Ticket-granting server
TGT:	Ticket-granting ticket
UI:	User interface
UMA:	User-Managed Access
URM:	User-Relationship Management
UUID:	Universally Unique Identifier.

Data Availability

The results rely on open specifications (mostly from the Internet Engineering Task Force—IETF) and available publications. All the hyperlinks appearing in the references or in the text have been checked prior to manuscript submission. Additionally, the academic and industrial software solutions studied in this article are free software, whose source code can be obtained freely on each solution's website as appearing in the references.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] N. Sakimura, J. Bradley, M. Jones, B. De Medeiros, and C. Mortimore, "Openid connect core 1.0 incorporating errata set 1," https://openid.net/specs/openid-connect-core-1_0.html Technical Report, OpenID, San Ramon, CA, USA, 2014, https://openid.net/specs/openid-connect-core-1_0.html Technical Report.
- [2] D. Hardt, "The OAuth 2.0 authorization framework," <https://rfc-editor.org/rfc/rfc6749.txt> Technical Report 6749, Internet Engineering Task Force, Fremont, CA, USA, 2012, <https://rfc-editor.org/rfc/rfc6749.txt> Technical Report 6749.

- [3] N. Kaaniche, M. Laurent, and S. Belguith, "Privacy enhancing technologies for solving the privacy-personalization paradox: taxonomy and survey," *Journal of Network and Computer Applications*, vol. 171, Article ID 102807, 2020.
- [4] M. Ates, S. Ravet, A. Mohamat Ahmat, and J. Fayolle, "An identity-centric Internet: Identity in the cloud, identity as a service and other delights," <https://ieeexplore.ieee.org/document/6045976> Technical Report, IEEE, Vienna, Austria, 2011, <https://ieeexplore.ieee.org/document/6045976> Technical Report.
- [5] Y.-A. de Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland, "OpenPDS: protecting the privacy of metadata through safe answers," *PLoS One*, vol. 9, no. 7, Article ID e98790, 2014.
- [6] E. Papadopoulou, A. Stobart, N. K. Taylor, and M. H. Williams, *Enabling Data Subjects to Remain Data Owners*, Springer International Publishing, Cham, Switzerland, 2015.
- [7] H. Haddadi, H. Howard, A. Chaudhry, J. Crowcroft, A. Madhavapeddy, and R. Mortier, "Personal data: thinking inside the box," 2015, <http://arxiv.org/abs/1501.04737>.
- [8] Entr'ouvert, "Fargo document manager presentation page," 2021, <https://dev.entrouvert.org/projects/fargo>.
- [9] D. Nuñez and I. Agudo, "Blindidm: a privacy-preserving approach for identity management as a service," *International Journal of Information Security*, vol. 13, no. 2, pp. 199–215, 2014.
- [10] Entr'ouvert, "The authentic 2 identity manager presentation page," 2021, <https://dev.entrouvert.org/projects/authentic>.
- [11] ForgeRock, "Openidm (documentation)," <https://backstage.forgerock.com/docs/openidm> Technical Report, ForgeRock, San Francisco, CA, USA, 2021, <https://backstage.forgerock.com/docs/openidm> Technical Report.
- [12] The OpenStack Foundation, "Keystone, the openstack identity service (documentation)," <https://docs.openstack.org/keystone/pike/> Technical Report, OpenStack, Austin, TX, USA, 2021, <https://docs.openstack.org/keystone/pike/> Technical Report.
- [13] R.H.. Keycloak, "Iam Documentation," Technical Report, RedHat, Raleigh, NC, USA, 2021.
- [14] C. Jan and B. Pfitzmann, *Federated Identity Management*, Springer, Berlin, Heidelberg, 2007.
- [15] C. Paquin, "U-prove technology overview v1.1 (revision 2)," 2013, <https://www.microsoft.com/en-us/research/publication/u-prove-technology-overview-v1-1-revision-2/>.
- [16] E. Maler, M. Machulak, J. Richer, and T. Hardjono, "User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization," Internet Engineering Task Force, Fremont, CA, USA, 2019, <https://datatracker.ietf.org/doc/html/draft-maler-oauth-umagrant-00> Internet-Draft draft-maler-oauth-umagrant-00.
- [17] A. Ceccanti, M. Hardt, B. Wegh, and A. Paul Millar, "The indigo-datacloud authentication and authorization infrastructure," *Journal of Physics: Conference Series*, vol. 898, no. 10, Article ID 102016, 2017.
- [18] European Parliament, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 4 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation)," 2016.
- [19] P. Marillonnet, M. Ates, M. Laurent, and N. Kaaniche, "An identity-matching process to strengthen trust in federated-identity architectures," in *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications*, vol. 3, pp. 142–154, SciTePress, Paris, France, July 2020.
- [20] Organization for the Advancement of Structured Information Standards, "Security assertion markup language (Saml) v2.0," <http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html> Technical Report, OASIS, Burlington, MA, USA, 2005, <http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html> Technical Report.
- [21] Roy Thomas Fielding, "REST: architectural styles and the design of network-based software architectures," <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm> Doctoral dissertation, University of California, Berkeley, CA, USA, 2000, <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm> Doctoral dissertation.
- [22] J. Reschke, "The "basic" HTTP authentication scheme," <https://rfc-editor.org/rfc/rfc7617.txt> Technical Report 7617, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7617.txt> Technical Report 7617.
- [23] C. Neuman, S. Hartman, K. Raeburn, and Y Taylor, "The kerberos network authentication service (V5)," <https://rfc-editor.org/rfc/rfc4120.txt> Technical Report 4120, Internet Engineering Task Force, Fremont, CA, USA, 2005, <https://rfc-editor.org/rfc/rfc4120.txt> Technical Report 4120.
- [24] M. Wolf and C. Wicksteed, "Date and time formats," <https://www.w3.org/TR/NOTE-datetime> Technical Report, W3C, Cambridge, MA, USA, 1997, <https://www.w3.org/TR/NOTE-datetime> Technical Report.
- [25] C. Newman and K. Graham, "Date and time on the internet: timestamps," Technical Report 3339 URL <https://rfc-editor.org/rfc/rfc3339.txt>, Internet Engineering Task Force, Fremont, CA, USA, 2002.
- [26] H. Schulzrinne, "The tel URI for telephone numbers," <https://rfc-editor.org/rfc/rfc3966.txt> Technical Report 3966, Internet Engineering Task Force, Fremont, CA, USA, 2004, <https://rfc-editor.org/rfc/rfc3966.txt> Technical Report 3966.
- [27] Kantara Consent & Information Sharing Work Group, "Consent receipt specification," <https://kantarainitiative.org/file-downloads/consent-receipt-specification-v1-1-0/> Technical Report, Kantara Initiative, Lonavla, India, 2018, <https://kantarainitiative.org/file-downloads/consent-receipt-specification-v1-1-0/> Technical Report.
- [28] M. Hodder, M. Lizar, M. Sabadello, and J. Wunderlich, "Minimum viable consent receipt (Mvcr) specification v.05," <https://kantarainitiative.org/confluence/display/archive/Minimum+Viable+Consent+Receipt+%28MVCR%29+Specification+v.05> Technical Report, Kantara Initiative, Lonavla, India, 2014, <https://kantarainitiative.org/confluence/display/archive/Minimum+Viable+Consent+Receipt+%28MVCR%29+Specification+v.05> Technical Report.
- [29] R. Shekh-Yusef, D. Ahrens, and S. Bremer, "HTTP digest access authentication," <https://rfc-editor.org/rfc/rfc7616.txt> Technical Report 7616, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7616.txt> Technical Report 7616.
- [30] J. Richer, M. Jones, J. Bradley, M. Machulak, and P. Hunt, "OAuth 2.0 dynamic client registration protocol," <https://rfc-editor.org/rfc/rfc7591.txt> Technical Report 7591, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7591.txt> Technical Report 7591.
- [31] B. Campbell, C. Mortimore, M. Jones, and Y. Yaron, "Goland. assertion framework for OAuth 2.0 client authentication and authorization grants," <https://rfc-editor.org/rfc/rfc7521.txt> Technical Report 7521, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7521.txt> Technical Report 7521.

- [32] B. Campbell, C. Mortimore, and M. Jones, "Security assertion markup language (SAML) 2.0 profile for OAuth 2.0 client authentication and authorization grants," <https://rfc-editor.org/rfc/rfc7522.txt> Technical Report 7522, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7522.txt> Technical Report 7522.
- [33] M. Jones, J. Bradley, and N. Sakimura, "JSON web token (JWT). RFC 7519," 2015, <https://rfc-editor.org/rfc/rfc7519.txt>.
- [34] C. Scott, J. Moreh, R. Philpott, and E. Maler, "Metadata for the oasis security assertion markup language (Saml) V2. 0," <https://docs.oasis-open.org/security/saml/v2.0/saml-metadata-2.0-os.pdf> Technical Report, OASIS, Burlington, MA, USA, 2005, <https://docs.oasis-open.org/security/saml/v2.0/saml-metadata-2.0-os.pdf> Technical Report.
- [35] M. Jones, N. Anthony, B. Campbell, J. Bradley, and C. Mortimore, "OAuth 2.0 token exchange," <https://rfc-editor.org/rfc/rfc8693.txt> Technical Report 8693, Internet Engineering Task Force, Fremont, CA, USA, 2020, <https://rfc-editor.org/rfc/rfc8693.txt> Technical Report 8693.
- [36] J. Richer, M. Jones, J. Bradley, and M. Machulak, "OAuth 2.0 dynamic client registration management protocol," <https://rfc-editor.org/rfc/rfc7592.txt> Technical Report 7592, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7592.txt> Technical Report 7592.
- [37] J. Richer, "OAuth 2.0 token introspection," <https://rfc-editor.org/rfc/rfc7662.txt> Technical Report 7662, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7662.txt> Technical Report 7662.
- [38] P. Hunt, G. Kelly, E. Wahlstroem, and C. Mortimore, "System for cross-domain identity management: core schema," <https://rfc-editor.org/rfc/rfc7643.txt> Technical Report 7643, Internet Engineering Task Force, Fremont, CA, USA, 2015, <https://rfc-editor.org/rfc/rfc7643.txt> Technical Report 7643.
- [39] M. Jones and D. Hardt, "The OAuth 2.0 authorization framework: bearer token usage," Internet Engineering Task Force, 2012, <https://rfc-editor.org/rfc/rfc6750.txt> Technical Report 6750.

Research Article

Privacy-Preserving Two-Factor Key Agreement Protocol Based on Chebyshev Polynomials

Zuowen Tan 

Department of Computing Science and Technology, Jiangxi University of Finance & Economics, Nanchang 330032, China

Correspondence should be addressed to Zuowen Tan; tanzyw@163.com

Received 24 December 2020; Revised 28 April 2021; Accepted 24 May 2021; Published 3 June 2021

Academic Editor: Ahmad Samer Wazan

Copyright © 2021 Zuowen Tan. 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.

Two-factor authentication is one of the widely used approaches to allow a user to keep a weak password and establish a key shared with a server. Recently, a large number of chaotic maps-based authentication mechanisms have been proposed. However, since the Diffie-Hellman problem of the Chebyshev polynomials defined on the interval $[-1, +1]$ can be solved by Bergamo et al.'s method, most of the secure chaotic maps-based key agreement protocols utilize the enhanced Chebyshev polynomials defined on the interval $(-\infty, +\infty)$. Thus far, few authenticated key agreement protocols based on chaotic maps have been able to achieve user unlinkability. In this paper, we take the first step in addressing this problem. More specifically, we propose the notions of privacy in authenticated key agreement protocols: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Then, we construct two two-factor authentication schemes with medium unlinkability based on Chebyshev polynomials defined on the interval $[-1, 1]$ and $(-\infty, +\infty)$, respectively. We do the formal security analysis of the proposed schemes under the random oracle model. In addition, the proposed protocols satisfy all known security requirements in practical applications. By using Burrows-Abadi-Needham logic (BAN-logic) nonce verification, we demonstrate that the proposed schemes achieve secure authentication. In addition, the detailed comparative security and performance analysis shows that the proposed schemes enable the same functionality but improve the security level.

1. Introduction

User authentication is indispensable for many information systems. Authenticated key agreement enables users to establish a session key which two or more parties share over a public channel. The session keys are adopted in subsequent secure communications. Password, hard-ware, and biometrics are always utilized in authentication mechanisms [1–3]. Single-factor authentication only provides limited security; then, the combination of these methods together can achieve higher security. Due to the convenient portability of smart cards, two-factor authentication [4–6] has been intensively investigated.

In general, user privacy protection during authenticated key exchange is a big challenge. For two-factor authentication schemes, the important issues should be addressed carefully. Firstly, the authentication mechanism should hide the user's identity from any eavesdroppers and foreign

servers. In other words, mutual authentication cannot reveal the real identity of the user, which is the basic goal of privacy protection, called anonymity. Secondly, the other aspect of user privacy protection is unlinkability. In many applications, the authentication mechanism should hide the user's movements from any eavesdroppers and other foreign servers. Any unauthorized entity cannot track the user's movements. Even if any outside adversary has accessed the message transmitted between the user and the server, it still cannot link the user to different authentication sessions.

Since chaotic maps provide the semigroup property and have higher efficiency than modular exponential operations and scalar multiplications on an elliptic curve, many chaotic, map-based, two-factor authenticated key agreement protocols [7–12] have been developed in recent years. Thus far, the user privacy preserving chaotic maps-based authenticated key agreement protocols have been intensively investigated [13–17].

1.1. Motivation. In two-factor authenticated key agreement protocols, the user privacy must be considered carefully. The basic security requirement is to preserve the user anonymity, while the stricter requirement is unlinkability or untraceability. These concepts in two-factor authenticated key agreement protocols are seldom discussed in detail. Till date, few two-factor authenticated key agreement protocols can provide users with a strong unlinkability.

To the best knowledge, most of the two-factor authenticated key agreements based on the Chebyshev polynomial protocols (hereafter, called TAKACP protocols) utilize the enhanced Chebyshev polynomials defined on the interval $(-\infty, +\infty)$. Now, few secure TAKACP protocols are based on the Chebyshev polynomials defined on the interval $[-1, +1]$. This is because those TAKACP protocols based on the Chebyshev polynomials over the interval $[-1, +1]$ are vulnerable to Bergamo et al.'s attacks [18].

1.2. Our Contributions. The main contribution of this paper is the two-factor authenticated key agreement protocol based on Chebyshev polynomials defined on the interval $[-1, 1]$ and $(-\infty, +\infty)$, respectively, which solve all above-mentioned issues for the first time. They satisfy more security requirements than the existing TAKACP protocols. In summary, we list the main contributions below:

The user privacy in two-factor authenticated key agreement protocol is expounded. According to the extent of user identity protection, the user privacy preserving in entity authentication protocols is classified into four concepts: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Of the four levels, the strong unlinkability is the highest while the anonymity-alone is fundamental for entity authentication. In this paper, we elaborate them by the formal probability model.

We analyze the Lin TAKACP protocol and reveal their weaknesses. Detailed analysis shows that it fails to provide session key security. And, it suffers from user impersonation attack. Next, it even cannot provide the weak unlinkability.

Analysis of formal security under Random Oracle model and BAN logic nonce verification demonstrate that the proposed schemes provide secure authentication against the CPCDH assumption and integer factorization hardness assumption.

The proposed schemes provide more security properties as compared to other TAKACP schemes. And the detailed comparative security analysis also shows that the proposed schemes avoid the weaknesses of the TAKACP protocols [19–21].

1.3. System Model

1.3.1. Network Model. The proposed two-factor authenticated key agreement protocols involve two entities, a user U and the server S . At the user registration phase, the server issues a smart card with secret information to U through a

secure channel. When the user U logs in to the server, the server and the user authenticate each other over the public channels. In this paper, the two-factor authenticated key agreement protocol is required to provide users with privacy preserving. Any adversary cannot trace the user from the message transmitted over open channels.

1.3.2. Adversary Model. Consider an adversary A who gets the full control over the communication channel between the user U and the service provider S (except the registration phase). Thus, A could obtain the messages transmitted between the user and the server (except the registration message). Of the phases in a two-factor authenticated key agreement protocol, only the registration phase requires a secure channel between U and S . In other phases, there could be various kinds of passive and active adversaries in the communication channel between U and S . The adversary A can eavesdrop and even block the message transmitted, modify messages, remove messages, or insert messages into the communication channel. Its objective is to compromise the mutual authentication between U and S . A even impersonates a user and attempts to access the server, or the adversary impersonates the server and provides the user with false services.

In the single-server environment, since users register on the same server with the same master key, the inside attacker A_0/A_1 is very powerful. Hereinafter, we refer to a malicious server which may try to recover the password of its client or track the client as an adversary A_0 ; a registered malicious user, or an adversary who has corrupted the user as an adversary A_1 ; and while other adversaries are called as outside adversary A_2 . To simulate the inside attack, A_0 and A_1 can get the passwords and information stored in the smart cards of the users except those of a client under attack. If the server is the attack target, A_1 is assumed to obtain the passwords and the information stored in the smart cards of all the users.

For a two-factor authentication scheme, the basic security property is that the user is required to both have the smart card and know the password. Since the smart cards cannot prevent the information stored in them from being extracted, for example, by monitoring their power consumption, the security of a two-factor authentication scheme is always discussed in the case that the smart card is stolen. In other words, when a user is under attack, A is allowed to either compromise the password or the smart card of the client under attack, but not both.

1.4. Organization of the Paper. The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces some preliminaries. Section 4 shows the limitations of the Lin protocol. Section 5 then presents two novel TAKACP protocols. Next, Section 6 analyzes the security of the proposed schemes. Comparison with the related smart-card-based protocols in terms of security properties and performance will be given in Section 7, and Section 8 is the conclusion.

TABLE 1: Cryptographic methodologies and drawbacks of the existing schemes and the proposed schemes.

	Fan et al. [22]	Juang et al. [23]	Sun et al. [24]	Li et al. [25]	Guo et al. [19]	Lin [20]	Lee [21]	Proposed TAKACP protocols
Cryptographic methodologies	Symmetric encryption Rabin's public-key cryptosystem	Symmetric encryption ECC	Symmetric encryption ECC	Public-key cryptosystem ECC	Symmetric encryption chaotic map on the interval $[-1,1]$	Symmetric encryption chaotic map on the interval $[-1,1]$	Symmetric encryption chaotic map on the interval $(-\infty,+\infty)$	Symmetric encryption, Rabin's public-key cryptosystem chaotic map on the interval $[-1,1]/(-\infty,+\infty)$
Drawbacks	Insecure, no privacy preservation	Insecure, no privacy preservation	Traceability	Insecure, no privacy preservation	Traceability	Traceability	No free password updating	None

2. Related Work

In this section, we briefly review some prior related works. Recent years have witnessed efforts on two-factor authentication [19–42]. We summarize some existing two-factor authentication schemes with their methodologies used, limitations, and drawbacks in Table 1.

2.1. Two-Factor Authentication Based on Enhanced Chebyshev Polynomials Defined on the Interval $(-\infty, +\infty)$ and Their Limitations. Researchers have developed chaotic maps-based key agreement protocols which utilize the enhanced Chebyshev polynomials defined on the interval $(-\infty, +\infty)$. Xiao et al. [31] first presented a chaotic map-based authenticated key agreement protocol by utilizing the semigroup property of Chebyshev chaotic maps [32, 33]. Guo and Zhang [34] showed that the Xiao et al.'s protocol [31] cannot provide the contributory property of key agreement. A malicious server can predetermine the session key. Guo and Zhang presented an improved version [34]. However, Lee demonstrated that the Guo-Zhang protocol [34] is insecure against off-line password guessing attacks [35]. In addition, it fails to provide the session key security. Tseng et al. [7] proposed anonymous key agreement protocol based on Chebyshev chaotic maps. Unfortunately, Niu et al. [8] demonstrate that Tseng et al.'s protocol [7] fails to protect the user anonymity and suffers from inside attacks. Yoon [9] found that the Niu-Wang protocol [8] is vulnerable to Denial of Service attacks. Xue et al. [36] also improved Tseng et al.'s protocol. However, Tan [37] pointed out that the Xue-Hong protocol [36] cannot still provide user anonymity. Moreover, the Xue-Hong protocol suffers from man-in-the-middle attacks. In 2012, Gong et al. [38] proposed password-based key agreement protocol by using extended chaotic maps. Unfortunately, Wang and Luan [39] showed that the key agreement protocol [38] suffers from potential security problems.

In 2014, Lin [40] developed an authentication scheme using dynamic identity and chaotic maps. Later, Islam et al. [41] state that Lin's scheme suffers from user impersonation attack. Islam et al. also presented an improved provably secure

scheme [41, 42] to solve the weaknesses of Lin's scheme. Unluckily, Jiang et al. [10] show that Islam's scheme is also vulnerable to some potential attacks. Based on the extended Chebyshev polynomials on the interval $(-\infty, +\infty)$, Lee et al. [21] presented improvement on Lin's scheme [20]. However, in the login phase of the improved scheme [21], the smart card fails to validate the input of the user. Moreover, in the password change phase, the server must participate in the whole updating process of each user. Hence, it is inconvenient for users to update the password in Lee et al.'s scheme [21].

2.2. Two-Factor Authentication Based on Enhanced Chebyshev Polynomials Defined on the Interval $[-1,+1]$ and Their Limitations. Few secure TAKACP protocols [19–21, 35] based on the Chebyshev polynomials defined on the interval $[-1,+1]$ have been presented.

In Lee's TAKACP scheme [35], the user and the server must preshare a password. Hence, when users register with the server, the server must share one different password with every user. In 2013, Guo and Chang [19] have proposed a smart-card-based authenticated key agreement using chaotic maps over the interval $[-1,+1]$. Subsequently, Hao et al. [43] and Lin [20] pointed out that there are some security pitfalls in the Guo-Chang scheme [19]. Any adversary can derive the session key only by using the messages transmitted between a user and the server. In addition, the Guo-Chang scheme fails to provide full protection for user identity due to a fixed parameter in every run of the protocol. To eliminate the above weaknesses, Lin presents an improved scheme [20] based on chaotic maps over the interval $[-1,+1]$. The Lin scheme is highly efficient since it is based on a simple symmetric cryptosystem. Unfortunately, Lee et al. [21] point out that the Lin scheme still fails to withstand denial-of-service and privileged insider attacks. In addition, the Lin scheme does not exhibit the contributory property of key agreements.

In this paper, we will show that the Lin scheme violates the session key security. The Lin scheme suffers from impersonation attacks. Specifically, it is still susceptible to Bergamo et al.'s attacks [44] from registered users of the

same server. Furthermore, we will demonstrate that the Lin scheme cannot provide the strong privacy protection. We also have found that it is inconvenient for users to update passwords in the Lin scheme [20] and the Guo–Chang scheme [19].

3. Mathematical Preliminaries

This section briefly introduces Chebyshev polynomials and two problems related to the chaotic maps. Then, we will discuss the user privacy in TAKACP protocols and define the different notions of user privacy.

3.1. Mathematical Preliminaries. Let n be an integer and x be a variable taking values over the interval $[-1,1]$. The Chebyshev polynomial $T_n: [-1,1] \rightarrow [-1,1]$ of degree n is defined as

$$T_n(x) = \cos(n \cdot \arccos(x)), \quad x \in [-1, 1]. \quad (1)$$

The recurrence relation of the Chebyshev polynomial is given by:

$$\begin{aligned} T_n(x) &= 2xT_{n-1}(x) - T_{n-2}(x), \quad n \geq 2, \\ T_0(x) &= 1, T_1(x) = x. \end{aligned} \quad (2)$$

The $\cos(s)$ and $\arccos(s)$ are defined as $\cos: R \rightarrow [-1, 1]$ and $\arccos: [-1, 1] \rightarrow [0, \pi]$. There are some examples of Chebyshev polynomials that are shown as follows:

$$\begin{aligned} T_2(x) &= 2x^2 - 1, \\ T_3(x) &= 4x^3 - 3x, \\ T_4(x) &= 8x^4 - 8x^2 + 1. \end{aligned} \quad (3)$$

The Chebyshev polynomials hold two important properties.

Semigroup property: Assume that r and s are positive numbers. For $x \in [-1, 1]$, $T_r(T_s(x)) = T_s(T_r(x))$. Due to its semigroup property, the Chebyshev polynomials satisfy the commutative property under the composition $T_r(T_s(x)) = T_s(T_r(x))$.

Chaotic property: Since the Chebyshev polynomial $T_n(x)$ with the positive integer n has a unique continuous invariant measure with positive Lyapunov exponent $\ln n$, it is a chaotic map with its invariant density $f^*(x) = 1/(\pi\sqrt{1-x^2})$. Specially, $T_2(x)$ is the well-known logistic map.

In 2008, Zhang [45] extended the definition of variables from the interval $[-1,1]$ to the interval $(-\infty, +\infty)$ as follows:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \bmod p, \quad n \geq 2, \quad (4)$$

where p is a large prime number. And, these enhanced Chebyshev polynomials still commute under the composition, $T_r(T_s(x)) = T_s(T_r(x)) \bmod p$.

The Chebyshev polynomials over the interval $(-\infty, +\infty)$ have the discrete logarithm problem and Diffie-Hellman

problem, which are assumed to be difficult to solve within a probabilistic polynomial time:

Definition 1. Chebyshev polynomial-based Discrete Logarithm (CPDL) problem. Given two elements x and y , find an integer r , such that $T_r(x) = y$, where $T_r(x)$ is the Chebyshev polynomial.

Definition 2. Chebyshev polynomial-based computational Diffie-Hellman (CPCDH) problem. Given three elements, x , Chebyshev polynomials $T_r(x)$, and $T_s(x)$, compute the value $T_{rs}(x)$.

In contrast with the Chebyshev polynomials over the interval $(-\infty, +\infty)$, the hardness assumption of CPCDH problems over the interval $[-1,1]$ does not hold. Given three elements, x , $T_r(x)$, and $T_s(x)$, although it is computationally infeasible to derive r from the known x and $T_r(x)$, one can apply the method mentioned in [18, 44] to derive

$$r^* = \frac{\arccos(T_r(x)) + 2k\pi}{\arccos(x)} | k \in Z, \quad (5)$$

such that $T_r(x) = T_{r^*}(x)$. Thus, one can compute the Diffie-Hellman value $T_{r^*}(T_s(x)) = T_s(T_{r^*}(x)) = T_s(T_r(x))$.

Definition 3. The success probability of a probabilistic polynomial time Turing machine Δ within time upper bound t in solving CPCDH problems is defined as:

$$\text{Adv}^{\text{CPCDH}}(t) = \Pr[\Delta(T_r(x), T_s(x)) = T_{rs}(x)]. \quad (6)$$

Definition 4. The Chebyshev polynomial-based computational Diffie-Hellman assumption (CPCDH assumption) is the assumption that CPCDH problems are hard. In other words, for every probabilistic Turing machine Δ , $\text{Adv}^{\text{CPCDH}}(t)$ is negligible.

Definition 5. Integer factorization assumption (IF assumption) is the assumption that integer factorization is hard. In other words, the probability $\text{Adv}^{\text{IF}}(t')$ of integer factorization for any probabilistic polynomial time Turing machine within the time upper bound t' is negligible.

3.2. Notions of Privacy in TAKACP Protocols. According to the extent of user identity protection, we divide user privacy preserving into four levels: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Among these concepts, the latter is stronger than the former, i.e., anonymity-alone \leq weak unlinkability \leq medium unlinkability \leq strong unlinkability.

Let N be the number of members of any user group. Let $A_{1,2}^{\text{Guess}}$ be the event that A (that is, A_1 and A_2 but A_0) guesses the identity of the user correctly from the user group, $A_2^{\text{Decide}}(A^{\text{Decide}})$ be the event that $A_2(A)$ decides whether any two executions of the protocol are from the same user, respectively.

Definition 6. (Anonymity-Alone). We define the advantage $\text{Adv}^{\text{Anon-Alone}}(A)$ of any adversary A as

$$\text{Adv}^{\text{Anon-Alone}}(A) = \left(\Pr[A_{1,2}^{\text{Guess}}] - \frac{1}{N} \right) + \left(\Pr[A_2^{\text{Decide}}] - 1 \right). \quad (7)$$

The advantage $\text{Adv}^{\text{Anon-Alone}}(A)$ measures the sum of the probability of any adversary A_1 or any outside adversary A_2 obtaining the identity of the user and the probability of any outside adversary A_2 linking different sessions with a certain user.

An authenticated key agreement protocol is called to provide Anonymity-Alone if $\text{Adv}^{\text{Anon-Alone}}(A)$ is negligible. In other words, for any group of N users, A cannot identify the actual user with the probability higher than the probability $1/N$ of guessing. Hence, the first addition item would approach 0. However, any outside adversary A_2 can link different sessions to a certain user.

Definition 7. (weak unlinkability). We define the advantage $\text{Adv}^{\text{Weak-Unlin}}(A)$ of any adversary A as

$$\text{Adv}^{\text{Weak-Unlin}}(A) = \left(\Pr[A_{1,2}^{\text{Guess}}] - \frac{1}{N} \right) + \left(\Pr[A_2^{\text{Decide}}] - \frac{1}{2} \right). \quad (8)$$

An authenticated key agreement scheme achieves weak unlinkability if $\text{Adv}^{\text{Weak-Unlin}}(A)$ is negligible. Specifically, any adversary A_1 or any outside adversary A_2 cannot obtain the identity of any other user. Besides, any outside adversary A_2 cannot link different sessions to a certain user with a probability larger than $1/2$.

Definition 8. (medium unlinkability). We define the advantage $\text{Adv}^{\text{Medium-Unlin}}(A)$ of any adversary A (here, A_1 and A_2 but A_0) as

$$\text{Adv}^{\text{Medium-Unlin}}(A) = \left(\Pr[A_{1,2}^{\text{Decide}}] - \frac{1}{2} \right). \quad (9)$$

An authenticated key agreement scheme is called to satisfy medium unlinkability if $\text{Adv}^{\text{Medium-Unlin}}(A)$ is negligible. In other words, any participant except the server cannot link different logins to the same user.

Definition 9. (strong unlinkability). We define the advantage $\text{Adv}^{\text{Strong-Unlin}}(A)$ of any adversary A including A_0 , A_1 , and A_2 as

$$\text{Adv}^{\text{Strong-Unlin}}(A) = \left(\Pr[A^{\text{Decide}}] - \frac{1}{2} \right). \quad (10)$$

An authenticated key agreement scheme is called to satisfy strong unlinkability if $\text{Adv}^{\text{Strong-Unlin}}(A)$ is negligible.

4. Cryptanalysis of the Lin Takacp Protocol

In this section, we first tabulate the important notations in Table 2. We then review Lin's key agreement protocol [20].

TABLE 2: The notations.

Symbol	Description
S	The server
U	A user
PW	U's password
ID	U's identity
s	The master key of the server
x	A variable with value in $[-1,1]$
\oplus	Exclusive-OR operation
$h()$	A one-way hash function
\parallel	String concatenation
T_1, T_2, T_3	The timestamps
t	A random integer
SK	Session key
$E_k()$	Symmetric encryption algorithm with k
$D_k()$	Symmetric decryption algorithm with k

4.1. Brief Review of Lin's Takacp Protocol. The Lin's TAKACP scheme [20] is composed of four algorithms: system initialization, user registration, authenticated key exchange, and password change. The notations used in [20] are listed in Table 2. Figures 1–3 separately illustrate the phases of user registration, authenticated key exchange, and password change.

4.1.1. System Initialization. The server S selects a master key s . Then, S computes a Chebyshev polynomial of degree r , i.e., $T_r(x)$, where $x \in [-1,1]$, and chooses a one-way hash function $h()$ and a symmetric encryption function $E_k()$ with the secret key k . S keeps r secret.

4.1.2. User Registration Phase. A user registration procedure consists of the following steps.

Step 1. The user U selects an identity ID , a password PW , and a random integer t . U computes $H = h(PW \parallel t)$ and then sends the message $\{ID, H\}$ to the server via a secure channel.

Step 2. Upon receiving the registration request, the server S computes $R = E_s(ID \parallel H)$, $D = H \oplus (x \parallel T_r(x))$. Then, S writes $\{R, h(), E_k(), D\}$ to a smart card and sends the smart card to U via a secure channel.

Step 3. Upon receiving the smart card, U stores t into it.

4.1.3. Authenticated Key Exchange Phase. U first enters the identity ID and password PW . Then, the smart card runs the following steps on S :

Step 1. It chooses a random integer j and computes

$$\begin{aligned} (x \parallel T_r(x)) &= D \oplus h(PW \parallel t), \\ v &= T_j(T_r(x)), \\ Q &= h(ID \parallel H). \end{aligned} \quad (11)$$

Then, it issues the message $\{T_j(x), E_v(Q, R, T_1)\}$ to S .

Step 2. S computes $v = T_j(T_r(x))$ and decrypts $E_v(Q, R, T_1)$. Then, S checks the validity of the time

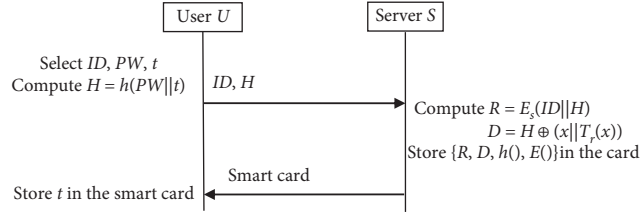


FIGURE 1: User registration phase of the Lin protocol.

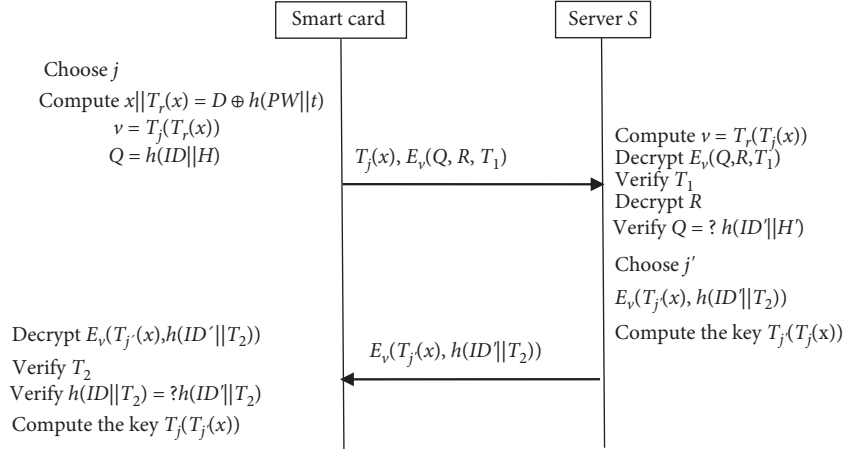


FIGURE 2: Authenticated key exchange phase of the Lin protocol.

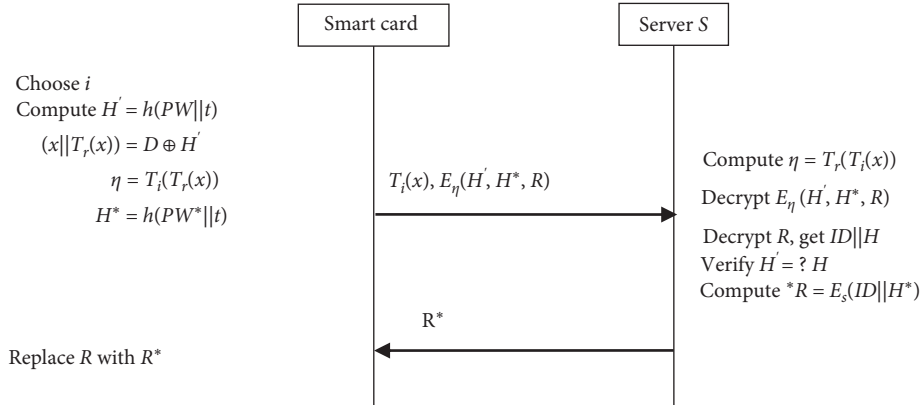


FIGURE 3: Password change phase of the Lin protocol.

stamp T_1 . Next, S decrypts R and verifies whether $Q = h(ID' || H')$ holds. If the equation holds, U is authenticated; otherwise, the session is terminated.

Step 3. S chooses a random integer j' and returns the login response $E_v(T_{j'}(x), h(ID' || T_2))$ to the card. And, S computes session key $T_{j'}(T_{j'}(x))$.

Step 4. The card first decrypts $E_v(T_{j'}(x), h(ID' || T_2))$ and then checks whether the delay time for T_2 is acceptable. Next, the card checks whether $h(ID || T_2) = h(ID' || T_2)$ holds. If the equation holds, S is authenticated. And, the card computes the session key $T_j(T_{j'}(x))$.

4.1.4. Password Change Phase. U first inserts the smart card into a terminal and inputs his or her identity ID , the old password PW , and a new password PW^* . Then, the smart card runs the following steps on S:

Step 1. The smart card chooses randomly a positive integer i and calculates

$$\begin{aligned}
 H' &= h(PW || t), \\
 (x || T_r(x)) &= D \oplus H', \\
 \eta &= T_i(T_r(x)), \\
 H^* &= h(PW^* || t),
 \end{aligned} \tag{12}$$

and delivers $\{T_i(x), E_\eta(H', H^*, R)\}$ to S.

Step 2. S computes $\eta = (T_r(T_i(x)))$, then decrypts $E_\eta(H', H^*, R)$ and further $R = E_s(ID||H)$. Next, S checks whether the received H' is equal to H . If the equation holds, the server returns $R^* = E_s(ID||H^*)$ to the card.

Step 3. The smart card replaces R with R^* .

4.2. Security Weaknesses of Lin's TAKACP Protocol. Lin [20] demonstrates that the Guo-Chang TAKACP scheme [19] cannot provide full protection for the user's identity. Any passive inside adversary A_1 (i.e., a malicious registered user) could derive the mutually shared session key between the user and the server only by intercepting the transmitted message. Lin claimed that their improvement eliminates the drawbacks. We show that the second security weakness of the Guo-Chang TAKACP scheme still exists in the Lin scheme [20]. In addition, the password change would not only bring inconvenience to the user but also lack the authentication of the server.

4.2.1. Violation of the Session Key Security. Assume that an inside adversary A_1 has intercepted the key exchange message transmitted between the user U and the server. In the Lin scheme, the adversary could derive the session key by performing the following steps.

Since A_1 is an inside adversary, A_1 can use his password to calculate $x||T_r(x)$. After intercepting U's login message $\{T_j(x), E_v(Q, R, T_1)\}$, A_1 has obtained $T_j(x)$. Although it is computationally infeasible to calculate j from x and $T_j(x)$, the adversary can apply the method mentioned in [18, 44] to compute

$$j^* = \frac{\arccos(T_j(x)) + 2k\pi}{\arccos(x)} | k \in Z, \quad (13)$$

such that $T_j(x) = T_{j^*}(x)$. Then, the adversary could compute the key $v = T_{j^*}(T_r(x)) = T_r(T_{j^*}(x)) = T_r(T_j(x))$.

A_1 decrypts $E_v(T_{j'}(x), h(ID||T_2))$ with the key v and obtains $T_{j'}(x)$. Finally, the adversary calculates the session key $SK = T_{j^*}(T_{j'}(x)) = T_{j'}(T_{j^*}(x)) = T_{j'}(T_j(x))$.

4.2.2. Suffering from User Impersonation Attack. Suppose that an inside adversary A_1 does not want to pay the server for the service provided by S. A_1 would try to impersonate a legitimate user U. After intercepting U's login message, the adversary launches the user impersonation attack as described below:

- (1) With the login message $\{T_j(x), E_v(Q, R, T_1)\}$, A_1 computes the key v through the same technique as in the leakage attack of the session key in Section 3. Then, A_1 decrypts $E_v(Q, R, T_1)$ and obtains $\{Q, R, T_1\}$.
- (2) A_1 chooses a random integer j and computes $v = T_j(T_r(x))$. It transmits $\{T_j(x), E_v(Q, R, T_1)\}$ to the server S, where T is the current timestamp.

- (3) After receiving the login message, S computes $v = T_r(T_j(x))$ and decrypts $E_v(Q, R, T_1)$. S first checks the validity of the time stamp T . Next, S decrypts R to derive $ID||H$, and verifies the identity. Since $Q = h(ID||H)$, the server believes that a legitimate user with ID has issued the login request. After that, S selects a Chebyshev polynomial $T_{j'}(x)$, encrypts it with other messages, and transmits the cipher-text to A_1 .
- (4) A_1 recovers the map $T_{j'}(x)$ with the key v and computes the session key.

4.2.3. Linking Different Sessions to a Same User. The Lin's TAKACP scheme enhances the protection of user identity. Although any adversary A_1 or A_2 cannot obtain the identity of any user, any inside adversary A_1 can link different sessions to a certain user. The Lin scheme only can provide weak unlinkability (for details, see Definition 7 in Section 3), since A_1 may execute the linking of sessions to the user as follows.

- (1) Intercept the different login messages $\{T_j(x), E_v(Q, R, T_1)\}$.
- (2) Compute the different keys v through the approach described in Section 3.
- (3) Decrypt $E_v(Q, R, T_1)$ and obtain $\{Q, R, T_1\}$. For the user, although the parameters $\{R, T_1\}$ change with different logins, Q will be unchanged. Thus, A_1 can decide whether the users are the same by comparing the parameter Q 's.

4.2.4. Defects of the Password Change. Firstly, the password change suffers from inside attacks. Consider that a registered user U' acts as an adversary A_1 . Since U' is a registered user of the server S, U' can derive $x||T_r(x)$ from his own (D', H') by using his own password. Assume that U' has intercepted U's password change request $\{T_i(x), E_\eta(H', H^*, R)\}$. As shown above, U' can calculate $i^* = ((\arccos(T_i(x)) + 2k\pi) / \arccos(x)) | k \in Z$, which satisfies the equation $T_i(x) = T_{i^*}(x)$. Thus, U' computes the key η and further decrypts $E_\eta(H', H^*, R)$. Then, U' selects randomly a value H'' of the same length of H^* . And U' computes $E_\eta(H', H'', R)$ and sends $\{T_i(x), E_\eta(H', H'', R)\}$ to the server. Since the equation $H' = H$ holds, the server will return $R^* = E_s(ID||H'')$ to the smart card. The smart card stores R^* instead of R . Thus, the password change has been fulfilled. However, the user U cannot login to the server any more with the new password PW^* . We describe the failure process as follows. When the user U tries to login to S, U computes $Q = h(ID||H^*)$ where $H^* = h(PW^*||t)$ and then issues $(T_j(x), E_v(Q, R^*, T_1))$ to the server. The server decrypts $E_v(Q, R^*, T_1)$ and acquires R^* . S further decrypts R^* to obtain $\{ID, H''\}$. S computes $Q' = h(ID||H'')$. Since $Q' \neq Q$, the server will refuse U's login request.

Secondly, the Lin's TAKACP scheme requires that the server participate during the whole password change phase. In many applications, registered users always need to update

their passwords at intervals. Passwords should be freely updated by the smart card holder at will without any interaction with the server, while the server can be totally unaware of the password change. A TAKACP scheme should provide the users with free password changes. If the users' password change requires the server online, it must be a bottleneck. The Lin scheme requires the server S to compute R^* during the password change phase. Therefore, it is inconvenient to both the server and the users. For the Lin scheme, the password change is impractical.

Thirdly, during the password change phase, the server is not authenticated by the smart card. This would be a serious security drawback. Any adversary could impersonate the server and send an arbitrary value as R^* . The smart card will replace R with R^* . The real card holder cannot login to the server any longer since $R^* \neq E_s(\text{ID} \| H^*)$. If the password change proceeds through a secure channel as in the user registration phase, the above security drawbacks will be removed. However, this is also infeasible.

5. The Proposed Takacp Protocol

Lin [20] showed that the Guo-Chang scheme suffers from inside attacks. The analysis above demonstrates that the Lin' TAKACP scheme [20] cannot still resist against inside attacks. The main cause is that an inside adversary A_1 has the common chaotic map $T_r(x)$ with the registered users of the same server. After intercepting the chaotic map $T_j(x)$ transmitted over the public channel, A_1 can derive an integer j^* satisfying $T_j(x) = T_{j^*}(x)$. The adversary computes the key v to decrypt $E_v(T_{j^*}(x), h(\text{ID} \| T_2), T_2)$ and derive $T_{j^*}(x)$. Thus, the adversary can determine the Diffie-Hellman-like session key $T_{j^*}(T_{j^*}(x))$.

To eliminate these weaknesses, we will seek cryptographic techniques to protect the functions $T_j(x)$ and $T_{j^*}(x)$. In the following, we will use the quadratic residues to present two improved versions. We first describe an improved two-factor authentication scheme (hereafter called TAKACP-1) based on Chebyshev polynomials defined on the interval $[-1,1]$ and then another two-factor authentication scheme (hereafter called TAKACP-2) based on Chebyshev polynomials defined on the interval $(-\infty, +\infty)$.

5.1. Registration Phase. We adopt the same notations as those in the Lin scheme. The server S selects s as the symmetric encryption key, two distinct large primes p and q with $p \equiv q \equiv 3 \pmod{4}$, and a one-way hash function $h(): \{0,1\}^* \rightarrow \{0,1\}^l$ where l is a security parameter. The parameters p , q , and s are kept secret. Before a user U gains access to the server S , U must register by performing the following steps as shown in Figure 4.

Step R1. U selects a random integer n' with l bits, an identity ID , and password PW . Then, U computes $d = h(\text{ID} \| PW)n'$ and delivers $\{ID, d\}$ to S through a secure channel.

Step R2. S computes $c = d \oplus E_s(\text{ID} \| n)$, where $n = pq$. S stores $\{c, n, h(\cdot)\}$ on a smart card and issues the smart card to U via a secure channel.

Step R3. The card computes.

5.2. Authenticated Key Exchange PHASE. The user U and the server S cooperatively perform the following steps to generate a session key SK , which is also illustrated in Figure 5.

Step A1. U inserts the smart card into the terminal and enters the identity ID and the password PW . The smart card checks whether d_2 is equal to $h(\text{ID} \oplus PW \| d_1 \| n)$. If ID and PW are valid, it computes $c = d_1 \oplus h(\text{ID} \| PW)$. It generates a nonce n_1 of the c 's binary length and randomly chooses a positive integer i and a real number x over $[-1,1]$. The card computes the Chebyshev polynomials $T_i(x)$, $e = (cn_1)^2 \pmod{n}$, $w_1 = h(n_1) \oplus (x \| T_i(x))$, $c_0 = h(\text{ID} \| n_1)$, $\theta_1 = h(c \| x \| T_i(x) \| c_0)$, where $x \| T_i(x)$ is of l -bits. The symbol means that the binary form of c interleaves the binary form of n_1 bit by bit. Then, the card transmits the message $M_1 = \{e, w_1, \theta_1\}$ to the server S .

Step A2. Once receiving the login request, S uses Chinese Remainder theorem with p and q to solve the square roots of e . S parses the four roots into two parts, c', n'_1 , respectively. Then, S decrypts c' to obtain (ID^*, n^*) and determines the right root by checking if n^* is equal to n . Finally, S obtains the right root $(c' n'_1)$ and the identity ID^* . S computes $(x \| T_i(x)) = h(n'_1) \oplus w_1$, $c_0^* = h(\text{ID}^* \| n'_1)$, and checks if the received θ_1 equals $h(c' \| x \| T_i(x) \| c_0^*)$. If the equation holds, the server believes that the login comes from a registered user with the identity ID^* . S generates a nonce n_2 such that $n_2 \| T_j(x)$ is l -bit in length and randomly chooses a positive integer j . Then, S computes

$$\begin{aligned} w_2 &= h(\text{ID}^* \| n'_1) \oplus (n_2 \| T_j(x)), \\ SK &= h(\text{ID}^* \| T_i(x) \| T_j(x) \| n'_1 \| n_2 \| T_j(T_i(x))), \\ \theta_2 &= h(c_0^* \| (T_j(x) \| x) \oplus h(n_2) \| SK). \end{aligned} \quad (14)$$

And, S sends back the message $M_2 = \{w_2, \theta_2\}$ to U . Otherwise, S rejects the login request from U .

Step A3. Upon receipt of the response message from S , the smart card computes $(n_2 \| T_j(x)) = w_2 \oplus h(\text{ID} \| n_1)$, $SK^* = h(\text{ID} \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_j(T_i(x)))$. Next, it checks whether θ_2 equals $h(c \| (T_j(x) \| x) \oplus h(n_2) \| SK^*)$. If they are equal, the card authenticates the server. It computes $\theta_3 = h(SK^* \| n_1 \| n_2 \| c_0)$ and forwards the message $M_3 = \{\theta_3\}$ to the server S . Otherwise, U terminates the session.

Step A4. After receiving the confirmation message from the card, S checks if $h(SK \| n'_1 \| n_2 \| c_0^*)$ equals θ_3 . If they are equal, the user U with identity ID is authenticated. Moreover, S confirms the session key SK .

5.3. Password Change Phase. If the user U wants to update his password, U performs the following steps.

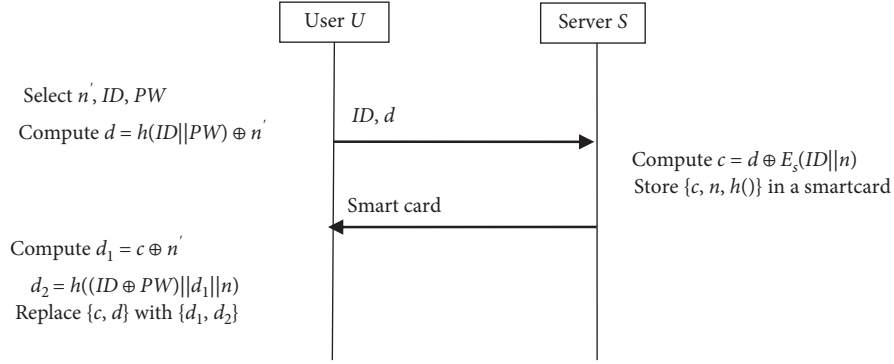


FIGURE 4: User registration phase of the proposed TAKACP-1 protocol.

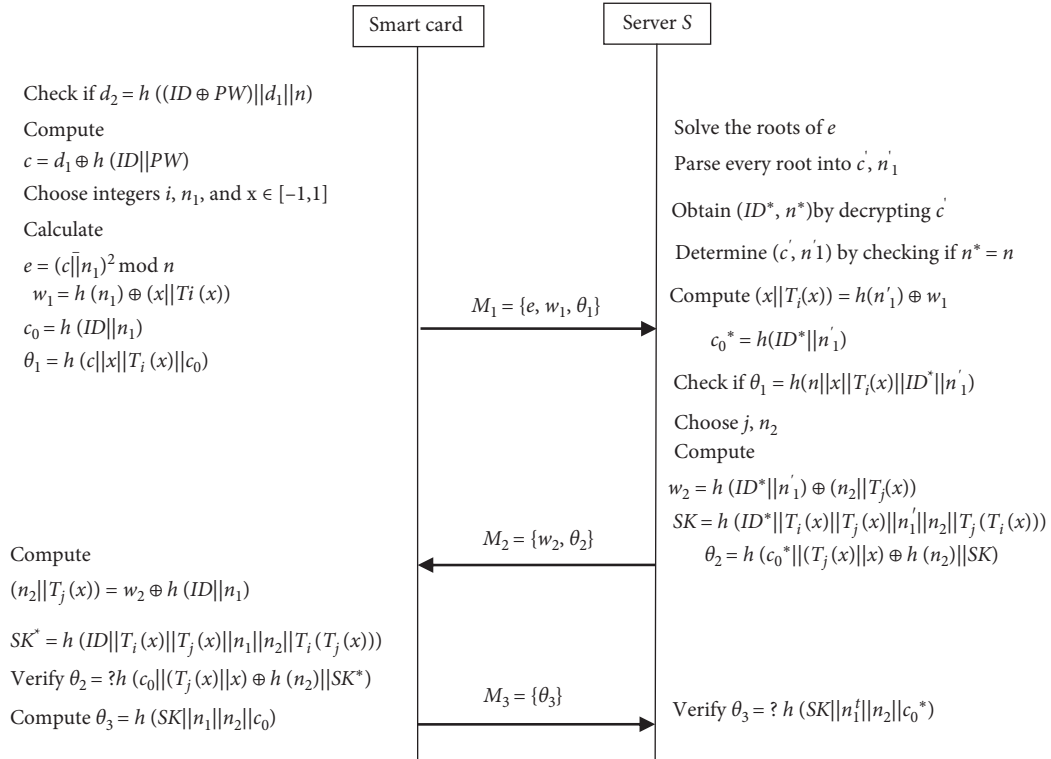


FIGURE 5: Authenticated key exchange phase of the proposed TAKACP-1 protocol.

Step C1. U inserts the smart card into the terminal and inputs the identity ID and the old password PW . Then, U issues the *updating* request.

Step C2. The smart card checks whether $d_2 = h((ID \oplus PW)||d_1||n)$ holds. If the equation holds, it answers *accepting updating*.

Step C3. U submits a new password PW_{new} .

Step C4. The smart card computes

$$\begin{aligned} d'_1 &= d_1 \oplus h(ID||PW) \oplus h(ID||PW_{\text{new}}), \\ d'_2 &= h((ID||PW_{\text{new}})d'_1||n). \end{aligned} \quad (15)$$

The card removes $\{d_1, d_2\}$ and stores $\{d'_1, d'_2\}$.

Now, we describe briefly the TAKACP-2 scheme. There is a little difference between the registration phase of the

TAKACP-2 scheme and that of the TAKACP-1 scheme. We will give the detailed description of the registration phase of TAKACP-2 scheme. Another fundamental difference of the authenticated key exchange phase of TAKACP-2 scheme from TAKACP-1 scheme is that the real number x is drawn from the interval $(-\infty, +\infty)$. The card/server computes the Chebyshev polynomial $T_i(x)/T_j(x) \bmod p_0$. By making similar modifications to the registration phase, we can have the password change phase. Here, we omit the description of the authenticated key exchange phase and the password change phase in the TAKACP-2 scheme.

5.4. Registration Phase of TAKACP-2. We adopt the same notations as those in the TAKACP-1 scheme. The server S selects a large prime p_0 besides the symmetric encryption key s , two large primes p, q , and a one-way hash function $h()$:

$\{0,1\}^* \longrightarrow \{0,1\}^l$. S keeps p , q , and s secret. Then, U performs the following steps to execute the registration.

Step R1'. U selects a random integer n' , an identity ID , and password PW . Then, U computes $d = h(ID\|PW)n'$ and delivers $\{ID, d\}$ to S through a secure channel.

Step R2'. S computes $c = d \oplus E_s(ID\|n\|p_0)$ where $n = pq$. S stores $\{c, n, p_0, h()\}$ on a smart card and issues the smart card to U via a secure channel.

Step R3'. The card computes

$$\begin{aligned} d_1 &= c \oplus n', \\ d_2 &= h((ID \oplus PW) \| d_1 \| (n \oplus p_0)). \end{aligned} \quad (16)$$

Then, U removes $\{n', c, d\}$ and stores $\{d_1, d_2\}$ in the card.

6. Security Analyses

In this section, we will present the formal semantic security analysis of the proposed protocols under the random oracle model in Part A. Mutual authentication between a user and a server will be confirmed through the widely used BAN logic [46–48] in Part B. In Part C, we conduct the detailed informal security analysis of the proposed protocol. The formal security analysis and informal security analysis both show that our schemes provide stronger security attributes.

6.1. Formal Security Analysis in Random Oracle Model. In this subsection, we introduce a formal security model under the widely used Real Or-Random model [49], the authentication security model [50], and the sequence of game models [51].

Assume that the server is a trustworthy entity. The server can accept the registration of users and validate the real identity of the users to provide them with services. There exists a secure channel between the server and the user to protect the registration of the user. In the following, we will apply the Dolev-Yao threat model (DY model) [52] to analyze the security of the proposed schemes. According to DY model, any two communicating parties communicate over an insecure channel. Assume that a polynomial time adversary has the ability to control the communication channel, for example, modifying, injecting, monitoring, and deleting messages over the open channel. Any adversary A can make oracle queries, which model the adversary's capabilities in a real attack. The goal of the adversary is to penetrate the anonymous authentication of a key agreement protocol by compromising requirements for the protocols described below. A malicious registered user may act as A to attempt to obtain the identity information of other users who have registered on the same server.

We will simulate various security attacks on the proposed schemes through all possible oracle queries listed below.

Execute(U^i, S^j): This query models eavesdropping attacks on honest execution among the user instance U^i and the server instance S^j . The output of this query consists of the messages that were exchanged during the honest execution of the protocol.

Send($U^i/S^j, m$): This query models an active attack. The oracle query enables A to receive an actual response from a participant U^i/S^j . Specifically, the adversary A sends a message m to instance U^i/S^j , and the participant instance U^i/S^j follows the protocol to give a reply.

Reveal(U^i/S^j): This query models known session key attacks. If no session key is defined for an instance U^i/S^j , or if either U^i/S^j , or its partner is asked a *Test* query, the output of this query is the invalid symbol \perp . Otherwise, it returns the current session key SK , which has been established between U^i/S^j and its partner to A .

Corrupt(U, a): The query models the capability of A to obtain the secret information of a user participant U , thereby corrupting the protocol.

If $a = 1$, the query returns U 's password to A .

If $a = 2$, the query returns the message stored in the user U 's smart card with A . The oracle simulates that when A gains the smart card of user U , it can extract the secret stored information.

Test(U^i/S^j): If no session key is defined, for instance, U^i/S^j or if either U^i/S^j or its partner is asked a *Reveal* query, the output of this query is the invalid symbol \perp . Otherwise, the oracle flips a coin b . If $b = 1$, the output is the session key. Otherwise, the output is a random string drawn from the space of session keys. The *Test* query is invoked once by the adversary with a fresh oracle. The query is used to define the semantic security of the session key SK .

Definition 10. An instance U^i/S^j is called to *be accepted*, if upon receiving the last expected protocol message, it goes into an accept state. The ordered concatenation of all sent and received messages by instance U^i/S^j forms the session identification(sid) of U^i/S^j for the current session.

Definition 11. Two instances U^i and S^j are said to *be partnered* if the three conditions hold simultaneously: (1) both U^i and S^j are accepted; (2) both U^i and S^j share the same sid; and (3) U^i and S^j are mutual partners of each other.

Definition 12. An instance U^i/S^j is called to *be fresh* if the following conditions are fulfilled simultaneously: (1) U^i/S^j is in the accepting state; (2) *Reveal(U^i/S^j)* query has never been submitted to U^i/S^j or its partner; and (3) strictly fewer than two *Corrupt(U^i, a)* queries have been made to U^i or strictly fewer than two *Corrupt(U^i, a)* queries have been submitted to S^j 's partner U^i .

Definition 13. For the *semantic security*, the security model is defined by a game, which consists of two phases. In the first phase, an adversary A is allowed to adaptively issue

Send, Execute, Reveal, and Test queries. In the second phase, the adversary A executes a single Test (U^i/S^i) query with the chosen bit b directed to a fresh instance and the query outputs a guess bit b' for b . If $b' = b$, then the adversary A wins the above game, i.e., A succeeds in breaking the semantic security of the game of a TAKACP protocol. Let $\text{Succ}(A)$ be an event where the adversary A wins the above game. The advantage of the adversary A in breaking the semantic security of the TAKACP protocol is defined by

$$\text{Adv}^{\text{TFAKA}}(A) = \left| \Pr\left(\text{Succ}(A) - \frac{1}{2}\right) \right| = \left| \Pr(b' = b) - \frac{1}{2} \right|. \quad (17)$$

A TAKACP protocol is said to be *semantically secure* if the advantage $\text{Adv}^{\text{TFAKA}}(A)$ of any probabilistic polynomial time-bounded adversary A is negligible.

Theorem 1. *Let $D(W)$ be a uniformly distributed password (identity) dictionary of size $|D|(|W|)$. Let A (including A_1 and A_2) be a polynomial time-bound adversary against the semantic security of the TAKACP-2 scheme. Suppose A makes at most q_s times Send queries, q_e times Execute queries, and q_h times hash oracle queries. Then, we have*

$$\begin{aligned} \text{Adv}^{\text{TFAKA-2}}(A) &\leq \frac{(q_s^2 + q_e^2)}{2^{n+1}} + \frac{(q_s^2 + q_e^2)}{2(p_0 - 1)} + \frac{q_h^2}{2^{l+1}} \\ &\quad + \frac{3q_h}{2^{l-1}} + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} + \frac{q_s}{2^{3l}} + \frac{q_s}{|D| + |W|} \\ &\quad + q_h \text{Adv}^{\text{CPCDH}}(t) + \text{Adv}^{\text{IF}}(t') \frac{q_s}{|W|}, \end{aligned} \quad (18)$$

where l refers to the string length of hash results, $\text{Adv}^{\text{CPCDH}}(t)$ is the success probability of any probabilistic polynomial time Turing machine within time upper bound t in solving CPCDH problems, and $\text{Adv}^{\text{IF}}(t')$ is the probability of integer factorization for any probabilistic polynomial time Turing machine within time upper bound t' .

Proof. We shall use the approach of sequent games to prove this theorem. We first define a sequence of modified attack games G_i ($i = 0, 1, 2, 3, 4, 5$) starting from G_0 and terminating at G_5 . Let Succ_i be an event defined as the successful guessing of the bit b in Test query corresponding to each game G_i by an adversary A .

Game G_0 : This starting game and the real protocol in random oracles are assumed to be identical. Hence, G_0 is the actual attack game. By definition, we have

$$\text{Adv}^{\text{TFAKA-2}}(A) = \left| \Pr[\text{succ}_0] - \frac{1}{2} \right|. \quad (19)$$

Game G_1 : This game is the same as the game G_0 except that the game simulates all oracle queries including Send, Reveal, Corrupt, Execute, Test, and hash queries. The hash oracles and Reveal, Test, Corrupt, and Execute

queries are simulated in Table 3. We simulate the Send queries in Table 4 as in the actual attack game. The simulations maintain three lists of queries: (1) list L_h records the answers to hash oracles, (2) list L_A records the answers to the queries which are initiated by A , and (3) list L_T records the transcripts between S and U .

This game is perfectly indistinguishable from the real execution of the protocol. Hence, we have

$$\Pr[\text{succ}_1] = \Pr[\text{succ}_0]. \quad (20)$$

Game G_2 : In this game, we consider collisions among the results of hash queries, random numbers, and Chebyshev polynomials in the transcripts of messages M_1, M_2 , and M_3 . We take the random value h from $\{0, 1\}^l$ as the response of the hash queries. If this query is directly asked by the adversary and $(*, h) \leftarrow L_h$, we abort the game. Otherwise, h is returned. Following the birthday paradox, the probability of collisions of the oracle hash query is at most $(qh^2/2^{l+1})$. Furthermore, the messages contain random numbers $\{n_1, n_2\}$ and two Chebyshev polynomials $\{T_i(x), T_j(x)\}$. And, the probability of random numbers and polynomials collision is at most $((q_s^2 + q_e^2)/2^{n+1}) + ((q_s^2 + q_e^2)/2(p_0 - 1))$. Games G_2 and G_1 are perfectly indistinguishable except that the abovementioned collision causes the game to abort. Hence, we have

$$|\Pr(\text{Succ}_2) - \Pr(\text{Succ}_1)| \leq \frac{(q_s^2 + q_e^2)}{2^{n+1}} + \frac{(q_s^2 + q_e^2)}{2(p_0 - 1)} + \frac{q_h^2}{2^{l+1}}. \quad (21)$$

Game G_3 : This game would abort the execution in the situation where A obtains a valid authenticator without active participation of hash oracles. In the TAKACP-2 protocol, the authenticated key exchange phase involves three message communications, $M_i, i = 1, 2, 3$. We consider three cases, $\text{Send}(U, M_1)$, $\text{Send}(S, M_2)$, $\text{Send}(U, M_3)$ in the game G_3 .

Case 1. Considering $\text{Send}(U, M_1)$ oracle query, we must carefully analyze the elements of message M_1 . The hash values $h(c\|x\|T_i(x)\|c_0) \in L_A$ must hold, otherwise the session will be terminated. The maximum calculated probability is up to $(q_h/2^l)$. Again, it must be that $h(ID\|n_1) \in L_A$ whose probability is at most $(q_h/2^l)$. Finally, the message $M_1 \in L_T$ should hold, or the session will stop. For this, the probability is $(q_s/2^{3l})$.

Case 2. Considering $\text{Send}(S, M_2)$ oracle query, M_2 consists of w_2 and θ_2 . The hash values $h(c_0^* \parallel (T_j(x) \oplus h(n_2)) \parallel h(ID^* \parallel T_i(x) \parallel T_j(x) \parallel n_1' \parallel n_2 \parallel T_j(T_i(x)))) \in L_A$ must hold; otherwise, the session will be terminated. The maximum probability is up to $(q_h/2^l)$. The probability of value $h(ID^* \parallel T_i(x) \parallel T_j(x) \parallel n_1' \parallel n_2 \parallel T_j(T_i(x)))$ falling within the list L_A is at most $(q_h/2^l)$. Finally, the message M_2 should fall within L_T , or the session will terminate. The maximum probability is $(q_s/2^{2l})$.

TABLE 3: Simulation of hash, reveal, test, corrupt, and execute oracle queries.

(i) *Hash* simulation query performs as follows:
If the record $(*; h)$ is found in the list L_h corresponding to the hash query h^* , return the hash function h . Otherwise, select a string $h \in \{0, 1\}^l$ and add $(*; h)$ into L_h . If the query is initiated by A , $(*; h)$ is stored in L_A .

(ii) *Reveal*(U^i/S^j) simulation query performs as follows:
If U^i/S^j is in the accepting state, the current session key SK formed by U^i/S^j and its partner is returned.

(iii) *Test*(U^i/S^j) simulation query performs as follows:
Through *Reveal*(U^i/S^j) query, obtain the current session SK and then flip an unbiased coin b . If $b = 1$, return SK . Otherwise, return a random string from $\{0, 1\}^l$.

(iv) *Corrupt*(U, a) simulation query performs as follows:
If $a = 1$, the query returns the password PW of U . If $a = 2$, the query returns the secret information stored in the user smart card.

(v) Simulation of *Execute*(U^i, S^j) query occurs in succession with the simulation of *Send* queries as shown below.
 $\{e, w_1, \theta_1\} \leftarrow \text{Send}(U^i; \text{start})$, $\{w_2, \theta_2\} \leftarrow \text{Send}(S^j; \{e, w_1, \theta_1\})$ and $\{\theta_3\} \leftarrow \text{Send}(U^i; \{w_2, \theta_2\})$. Finally, $M_1 = \{e, w_1, \theta_1\}$, $M_2 = \{w_2, \theta_2\}$, and $M_3 = \{\theta_3\}$ are returned.

TABLE 4: Simulation of *send* oracle queries.

(i) On a query $\text{Send}(U^i; \text{start})$, assuming U^i is in the correct state, we proceed as follows:
Choose a positive integer i and a real number x over $(-\infty, +\infty)$ and compute $T_i(x)$, $e = (cn_1)^2 \bmod n$, $w_1 = h(n_1) \oplus (x \| T_i(x))$, $\theta_1 = h(c \| x \| T_i(x) \| h(ID \| n_1))$. Then, the answer $\{e, w_1, \theta_1\}$ to the query is returned.

(ii) On a query $\text{Send}(S^j; \{e, w_1, \theta_1\})$, assuming S^j is in the correct state, we proceed as follows:
Solve the square roots of e and obtain c', n'_1 . Decrypt c' and get (ID^*, n^*) . Compute $(x \| T_i(x)) = h(n'_1) \oplus w_1$, $c_0^* = h(ID^* \| n'_1)$, and check if the received $\theta_1 = h(c' \| x \| T_i(x) \| c_0^*)$. If the equation does not hold, the server instance terminates without accepting. Otherwise, choose randomly a nonce n_2 , a positive integer j , and compute $w_2 = h(ID^* \| n'_1) \oplus (n_2 \| T_j(x))$, $SK = h(ID^* \| T_i(x) \| T_j(x) \| n'_1 \| n_2 \| T_j(T_i(x)))$, $\theta_2 = h(c_0^* \| (T_j(x) \oplus h(n_2)) \| SK)$. Then, the answer $\{w_2, \theta_2\}$ to the query is returned.

(iii) On a query $\text{Send}(U^i; \{w_2, \theta_2\})$, assuming U^i is in the correct state, we proceed as follows:
Compute $(n_2 \| T_j(x)) = w_2 \oplus h(ID \| n_1)$, $SK^* = h(ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_i(T_j(x)))$, and check whether $\theta_2 = h(c \| (T_j(x) \oplus h(n_2)) \| SK^*)$. If the equation does not hold, the user instance terminates without accepting. Otherwise, compute $\theta_3 = h(SK^* \| (n_1 \oplus n_2) \| c_0)$, authenticate S^j , and establish SK as the session key. Then, the answer $\{\theta_3\}$ to the query is returned.

(iv) On a query $\text{Send}(S^j; \{\theta_3\})$, assuming S^j is in the correct state, we proceed as follows:
Check if $\theta_3 = h(SK \| (n'_1 \oplus n_2) \| c_0^*)$. If the equation does not hold, the server instance terminates without accepting and aborts the session. Otherwise, S accepts the session key SK .

Case 3. To respond $\text{Send}(U, M_3)$ oracle query, $h(h(ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_i(T_j(x))) \| (n_1 \oplus n_2) \| c_0) \in LA$ must hold with the total maximum probability $(q_h/2^l)$. Finally, for a transcript $M_3 \in L_T$, we have the maximum probability as $(q_s/2^l)$.

Considering the three cases, we have,

Game G_4 : In this game, when the session key SK is required to compute, we replace the random hash oracle H_1 with private oracle H . That is, the session key is determined without querying the hash oracle. Moreover, we do not use $T_i(T_i(x))$ or $T_i(T_j(x))$ to compute $SK = H'(ID \| T_i(x) \| T_j(x) \| n_1 \| n_2)$. Thus, the session key is completely independent of *hash oracle* and $T_i(T_i(x))$ or $T_i(T_j(x))$. Games G_4 and G_3 are perfectly indistinguishable unless the following event AskH_1 occurs: the adversary A queries the hash function on $ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_j(T_i(x))$ or on the message $ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_i(T_j(x))$. Hence, we have

$$|\Pr(\text{Succ}_4) - \Pr(\text{Succ}_3)| \leq \Pr[\text{AskH}_1]. \quad (23)$$

Game G_5 : In this game, we simulate the executions using the random self-reducibility of the CPCDH problem, given one CPCDH instance $(T_i(x), T_j(x))$. We choose

randomly two integers u, v and compute $T_u(T_i(x)), T_v(T_j(x))$. The event AskH_2 means that the adversary A had queried the random hash oracle H_1 on $ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| Z$, where $Z = \text{CPCDH}(T_u(T_i(x)), T_v(T_j(x)))$. It is easy to know that the equation $\text{CPCDH}(T_u(T_i(x)), T_v(T_j(x))) = T_{uv}(\text{CPCDH}(T_i(x), T_j(x)))$ holds. We have

$$\begin{aligned} & |\Pr(\text{Succ}_5) - \Pr(\text{Succ}_4)|, \\ & \Pr[\text{AskH}_1] = \Pr[\text{AskH}_2]. \end{aligned} \quad (24)$$

According to the definition of the event AskH , the accumulated probability is at least $\Pr[\text{AskH}_2]/q_h$. Thus, we have

$$\Pr[\text{AskH}_2] \leq q_h \text{Adv}^{\text{CPCDH}}(t). \quad (25)$$

In Game G_5 , Diffie-Hellman keys SK are random and independent of passwords and ephemeral keys. So, there are two possible cases where the adversary distinguishes the real session key from the random key as follows:

Case 1. The adversary queries the hash oracle on $ID \| T_i(x) \| T_j(x) \| n_1 \| n_2 \| T_j(T_i(x))$. The probability that this event occurs is $(q_h/2^l)$.

Case 2. The adversary asks the *Send* (U^t, m) query and successfully impersonates a user. If the *Corrupt*($U, 1$) has been made, it implies that the *Corrupt* ($U, 2$) has not been made. To impersonate the user, the adversary has to obtain the parameter c and the identity. The probability that the event happens is $\text{Adv}^{\text{IF}}(t')(q_s/|W|)$. On the contrary, if the *Corrupt* ($U, 2$) has been made, it is not allowed to reveal the static key PW of the user. Thus, in order to impersonate the user, the adversary has to obtain some information on the password of the user. The success probability of the adversary in the q_s sessions is $(q_s/(|D| + |W|))$. If the adversary just makes an attempt at random to impersonate the user by computing θ_3 and succeeds, it will make the difference; but, the probability for q_s sessions is less than. $(q_s/2^l)$

Hence, we have

$$\Pr[\text{Succ}_5] \leq \frac{1}{2} + \frac{q_h}{2^l} + \text{Adv}(t') \frac{q_s}{|W|} + \frac{q_s}{|D| + |W|} + \frac{q_s}{2^l}. \quad (26)$$

Using the triangular inequality and equations (19)–(26), we have the following:

$$\begin{aligned} \text{Adv}^{\text{TFAKA-2}}(A) &= \left| \Pr\left[\text{Succ}_0 - \frac{1}{2}\right] \right| \\ &\leq \sum_{i=0}^4 \left| \Pr[\text{Succ}_i] - \Pr[\text{Succ}_{i+1}] \right| \\ &\quad + \left| \Pr[\text{Succ}_5] - \frac{1}{2} \right| \\ &\leq \frac{(q_s^2 + q_e^2)}{2^{n+1}} + \frac{(q_s^2 + q_e^2)}{2(p_0 - 1)} + \frac{q_h^2}{2^{l+1}} + \frac{3q_h}{2^{l-1}} \\ &\quad + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} + \frac{q_s}{2^{3l}} + \frac{q_s}{|D| + |W|} \\ &\quad + q_h \text{Adv}^{\text{CPCDH}}(t) + \text{Adv}^{\text{IF}}(t') \frac{q_s}{|W|}. \end{aligned} \quad (27)$$

Thus, we have completed the proof of the theorem.

The above theorem is about the security of the proposed TAKACP-2 scheme based on the extended Chebyshev polynomials defined on the interval $[-\infty, +\infty]$. To complete the security proof of the TAKACP-1 scheme based on Chebyshev polynomials over the interval $[-1, 1]$, one only needs to delete the CPCDH simulation in Game G_5 of the proof of Theorem 1. We only state the results as Theorem 2 without detailed proof. \square

Theorem 2. *Let $D(W)$ be a uniformly distributed password (identity) dictionary of size $|D|(|W|)$. Let A (including A_1 and A_2) be the polynomial time-bound adversary against the semantic security of the TAKACP-1 scheme. Suppose A makes*

Send queries q_s times, Executes queries q_e times, and hash oracle queries q_h times at most. Then, we have

$$\begin{aligned} \text{Adv}^{\text{TFAKA-2}}(A) &\leq \frac{(q_s^2 + q_e^2)}{2^{n+1}} + \frac{q_h^2}{2^{l+1}} + \frac{3q_h}{2^{l-1}} + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} \\ &\quad + \frac{q_s}{2^{3l}} + \frac{q_s}{|D| + |W|} + \text{Adv}^{\text{IF}}(t') \frac{q_s}{|W|}, \end{aligned} \quad (28)$$

where l refers to the string length of hash results, $\text{Adv}^{\text{CPCDH}}(t)$ is the success probability of any probabilistic polynomial-time Turing machine within the time upper bound t in solving CPCDH problems, and $\text{Adv}^{\text{IF}}(t')$ is the probability of any probabilistic polynomial-time Turing machine in solving the square root with composite number module within time upper bound t' .

Theorem 3. *The proposed TAKACP-1(TAKACP-2)scheme achieves the property of the medium unlinkability.*

Proof. Consider that the insider adversary A_1 would attempt to violate the user anonymity of the proposed schemes. Further suppose that the *Corrupt* ($U, 2$) has been made. The smart card of the user U is compromised. The adversary A has extracted the elements $\{d_1, d_2, n\}$ for TAKACP-1($\{d_1, d_2, n, p_0\}$ for TAKACP-2) in the smart card. Since $d_1 = E_s(ID||n) \oplus h(ID||PW)$, $d_2 = h((ID \oplus PW)||d_1||n)$ in TAKACP-1, or $d_1 = E_s(ID||n||p_0) \oplus h(ID||PW)$, $d_2 = h((ID \oplus PW)||d_1||n \oplus p_0)$ in TAKACP-2, A cannot divide d_1 or d_2 into the exclusive-OR items $E_s(ID||n)$ or $E_s(ID||n||p_0)$, $h(ID||PW)$, $h(ID \oplus PW)$ correctly. In essence, even if A has $E_s(ID||n)$, A cannot still recover ID from it without the master key s . Since $d_2 = h((ID \oplus PW)||d_1||n)$ or $h((ID \oplus PW)||d_1||n \oplus p_0)$, owing to the one-way hash function, A_1 cannot derive ID or PW from d_2 .

Now consider the authenticated key exchange phase. Assume that $M_{0,1} = \{e_0, w_{0,1}, \theta_{0,1}\}$ and $M_{1,1} = \{e_1, w_{1,1}, \theta_{1,1}\}$ are two requesting messages produced by one user in two different authentication sessions, where

$$\begin{aligned} e_{0,1} &= (cn_{0,1})^2 \bmod n, \\ w_{0,1} &= h(n_{0,1}) \oplus (x||T_{0i}(x)), \\ \theta_{0,1} &= h(c||x||T_{0i}(x)||h(ID||n_{0,1})), \\ e_{1,1} &= (cn_{1,1})^2 \bmod n, \\ w_{1,1} &= h(n_{1,1}) \oplus (y||T_{1i}(y)), \\ \theta_{1,1} &= h(c||y||T_{1i}(y)||h(ID||n_{1,1})). \end{aligned} \quad (29)$$

Due to the usage of the random integers $\{n_{0,1}, n_{1,1}\}$, $e_{0,1}$ is independent of $e_{1,1}$. Likewise, owing to the randomness of $\{x, y, \theta_{0,1}, \theta_{1,1}\}$, $T_{0i}(x)$ is independent of $T_{1i}(y)$. Thus, $w_{0,1}$ is independent of $w_{1,1}$. As $h()$ is a secure cryptographic hash function, the same is true for $\theta_{0,1}$ and $\theta_{1,1}$. Therefore, A believes that $M_{0,1}$ and $M_{1,1}$ are independent of each other. We can make similar analysis of the response messages $M_{0,2} = \{w_{0,2}, \theta_{0,2}\}$, $M_{1,2} = \{w_{1,2}, \theta_{1,2}\}$, and the confirmation

messages $M_{0,3} = \{\omega_{0,3}, \theta_{0,3}\}$, $M_{1,3} = \{\omega_{1,3}, \theta_{1,3}\}$. From the above analysis, we have $\Pr[A^{\text{Decide}}] = 1/2$. Thus, $\text{Adv}^{\text{Strong-Unlink}}(A) = 0$.

Therefore, our protocols achieve medium unlinkability. Any adversary (but A_0) is unable to link two different protocol sessions to the same user. \square

6.2. Authentication Proof Based on BAN-Logic. In this section, we introduce the well-popular Burrows-Abadi-Needham Logic (BAN-logic) to validate the authentication of the proposed protocols. By using BAN logic, we also try to find out flaws in the proposed schemes and deal with authentication issues among the participants. The formal verification of the BAN logic demonstrates that the proposed protocols achieve mutual authentication and allow the user and the server to establish session keys. It is well-known that BAN logic [46] is the widely used logical analysis method of reasoning the beliefs of participants in an authentication protocol [47, 48]. BAN logic uses a set of postulates to analyze and verify authentication schemes. BAN logic has three elementary items, i.e., formulas/statements, principals, and keys. Let X and Y be two statements, P and Q be principals, K be the symbol for a key. The basic expressions of BAN logic are described in Table 5. More details can be found in [46–48].

The main logical postulates of the BAN logic are listed as follows:

Message-meaning_K rule: $(P \equiv P \stackrel{K}{\leftrightarrow} Q, P \triangleleft \{X\}_K / P \equiv Q \sim X)$, $(P \equiv P \stackrel{K}{\leftrightarrow} Q, P \triangleleft \langle X \rangle_K / P \equiv Q \sim X)$: If P believes that it shares K with Q and sees X encrypted by K (or X combined with K), then P believes that Q once said X .

The nonce-verification rule: $(P \equiv \#(X), P \equiv Q \sim X / P \equiv Q \equiv X)$

If P believes that X could have been uttered only recently and Q once said X , then P believes that Q believes X .

The freshness propagation rule: $(P \equiv \#(X) / P \equiv \#(X, Y))$

If P believes that X is fresh, then P also believes that (X, Y) is fresh.

The jurisdiction rule: $(P \equiv Q \Rightarrow X, P \equiv Q \equiv X / P \equiv X)$

If P believes that Q has authority over X and Q believes X , then P trusts Q on the truth of X .

The message decryption rule: $(P \equiv P \stackrel{K}{\leftrightarrow} Q, P \triangleleft \{X\}_K / P \triangleleft X)$

If P believes that it shares K with Q and sees encrypted X by K , then P sees X .

In the following, we apply BAN logic to analyze the TAKACP-1 scheme. Similar analysis can be applied to the TAKACP-2 scheme. According to the analytic procedures of the BAN logic, the proposed TAKACP-1 scheme must satisfy the following goals:

$$\text{Goal (1): } U \equiv S \equiv U \stackrel{\text{SK}}{\leftrightarrow} S,$$

$$\text{Goal (2): } U \equiv U \stackrel{\text{SK}}{\leftrightarrow} S,$$

$$\text{Goal (3): } S \equiv U \equiv U \stackrel{\text{SK}}{\leftrightarrow} S,$$

$$\text{Goal (4): } S \equiv U \stackrel{\text{SK}}{\leftrightarrow} S.$$

The generic form of the proposed TAKACP-1 scheme is described below.

From message M_1 , $U \rightarrow S: \{c, n_1\}_n, \langle x, T_i(x) \rangle_{h(n_1)}, c_0, x, T_i(x) \rangle_c$.

From message M_2 , $S \rightarrow U: \langle n_2, T_j(x) \rangle_{h(\text{ID}^* \| n_1)}, \langle T_j'(x), h(n_2), \text{SK} \rangle_{c_0}$.

From message M_3 , $U \rightarrow S: \langle n_1 \oplus n_2, \text{SK} \rangle_{c_0}$.

The idealized form of the proposed TAKACP-1 scheme in the language of formal logic is given below.

$$\text{Message } M_1: U \rightarrow S: \langle x, T_i(x), U \stackrel{c_0}{\leftrightarrow} S, U \stackrel{n}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

$$\text{Message } M_2: S \rightarrow U: \langle n_2, T_j(x), U \stackrel{\text{SK}}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

$$\text{Message } M_3: U \rightarrow S: \langle n_1, n_2, U \stackrel{\text{SK}}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

We make the assumptions about the initial state to analyze the proposed TAKACP-1 scheme:

$$H_1: U \equiv \#(n_1), U \equiv \#(T_i(x));$$

$$H_2: S \equiv \#(n_2), S \equiv \#(T_j(x));$$

$$H_3: S \equiv U \stackrel{c_0}{\leftrightarrow} S;$$

$$H_4: U \equiv U \stackrel{c_0}{\leftrightarrow} S;$$

$$H_5: S \equiv U \Rightarrow U \stackrel{c_0}{\leftrightarrow} S;$$

$$H_6: U \equiv S \Rightarrow U \stackrel{\text{SK}}{\leftrightarrow} S;$$

$$H_7: S \equiv U \Rightarrow U \stackrel{\text{SK}}{\leftrightarrow} S.$$

According to the BAN logic and the assumptions, we give the main proof as follows:

From message M_1 , we have

$$S_1: S \triangleleft \langle x, T_i(x), U \stackrel{c_0}{\leftrightarrow} S, U \stackrel{n}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

From S_1 , H_3 , and Rule (1), we have

$$S_2: S \equiv U \sim \langle x, T_i(x), U \stackrel{c_0}{\leftrightarrow} S, U \stackrel{n}{\leftrightarrow} S \rangle.$$

From S_2 , H_2 , Rule (2) and Rule (3), we have

$$S_3: S \equiv U \equiv U \stackrel{c_0}{\leftrightarrow} S.$$

From S_3 , H_5 , and Rule (4), we have

$$S_4: S \equiv U \stackrel{c_0}{\leftrightarrow} S.$$

From message M_2 , we have

$$S_5: U \triangleleft \langle n_2, T_j(x), U \stackrel{\text{SK}}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

From S_5 , H_4 , and Rule (1), we have

$$S_6: U \equiv S \sim \langle n_2, T_j(x), U \stackrel{\text{SK}}{\leftrightarrow} S \rangle.$$

From S_6 , H_1 , Rule (2), and Rule (3), we have

$$S_7: U \equiv S \equiv U \stackrel{\text{SK}}{\leftrightarrow} S(\text{Goal (1)}).$$

From S_7 , H_6 , and Rule (4), we have

$$S_8: U \equiv U \stackrel{\text{SK}}{\leftrightarrow} S(\text{Goal (2)}).$$

From message M_3 , we have

$$S_9: S \triangleleft \langle n_1, n_2, U \stackrel{\text{SK}}{\leftrightarrow} S \rangle_{U \stackrel{c_0}{\leftrightarrow} S}$$

From S_9 , S_4 , and Rule (1), we have

$$S_{10}: S \equiv U \sim \langle n_1, n_2, U \stackrel{\text{SK}}{\leftrightarrow} S \rangle.$$

TABLE 5: Notations of BAN logic.

Symbol	Describe
$P \equiv X$	The principal P believes a statement X , or P would be entitled to believe X .
$P \triangleleft X$	P sees X . P has received a message containing X and can read and repeat X (possibly after doing some decryption).
$P \sim X$	P once said X . P at some time sent a message containing X . It is not known whether this is a replay, though it is known that P believed X when he or she sent it.
$P \equiv X$	P has jurisdiction over X . P is an authority on X and is trusted on this matter.
$\#(X)$	The formula X is fresh. That is, X has never been sent in a message at any time before the current run of the protocol
$P \stackrel{K}{\leftrightarrow} Q$	K is a key shared between P and Q . P and Q may use K to communicate. And K is good since it can never be discovered by any principal except P or Q , or a principal trusted by either P or Q .
$P \stackrel{X}{\leftrightarrow} Q$	The formula X is a shared key known only to P and Q , possibly to principals trusted by them.
$\{X\}_K$	The formula X is encrypted by K .
$\langle X \rangle_Y$	This represents X combined with the formula Y . It is intended that Y be a secret and that its presence proves the identity of whoever utters $\langle X \rangle_Y$. X is simply concatenated with Y while Y plays a role as proof or origin for X .

From S_{10} , H_2 , Rule (2), and Rule (3), we have

$$S_{11}: S| \equiv U| \equiv U \stackrel{SK}{\leftrightarrow} S(\text{Goal } (3)).$$

From S_{11} , H_7 , and Rule (4), we have

$$S_{12}: S| \equiv U \stackrel{SK}{\leftrightarrow} S(\text{Goal } (4)).$$

6.3. Informal Security Analysis. In this section, we analyze the security of the proposed protocols. We will show that the proposed protocols satisfy the essential security requirements, including the ability to provide medium unlinkability, the contributory property of key agreements, session key security, two-factor secrecy, and free updating of passwords. Furthermore, we confirm that the proposed schemes can withstand replay attacks and password-guessing attacks. In the following, we will not expound the two-factor security since its proof has been given in Part A and Part B of Section 4, respectively.

6.3.1. Medium Unlinkability. In Part A of Section 4, we have demonstrated that our protocols can provide medium unlinkability. Now, we compare the privacy-preserving of our protocols with that of the schemes of Guo-Chang [19], Lin [20], and Sun et al. [24].

In the Guo-Chang scheme, since the user identity is encrypted with the master key of the server into the login request R , any adversary A cannot reveal the user identity. So $\Pr[A^{Guess}] = 1/N$. However, R is unchanged until the user updates the password. Thus, any outside adversary can distinguish whether the users in two authentication sessions are identical. That is, $\Pr[A_2^{Decide}] = 1$. Thus, we have $\text{Adv}^{\text{Anon-Along}}(A) = 0$ and $\text{Adv}^{\text{Weak-Unlin}}(A) \geq 1/2$. Hence, the Guo-Chang scheme provides *anonymity-alone* but *weak unlinkability*. Similarly, since every request of the user contains the unchanged element IM , Sun et al.'s scheme [24] only achieves the property *anonymity-alone*.

The Lin scheme can provide weak unlinkability. Specifically, although the elements Q and R are kept unchanged, they are transmitted in the ciphertext $E_v(Q, R, T_1)$. Since any outside adversary A_2 cannot calculate the key v , they cannot obtain R . But any inside adversary A_1 can compute v and acquire Q, R . For every login of the user, the parameters $\{Q, R\}$ are static until the user changes its password or identity. Thus, A_1 can still decide whether different login requests are

from the same user or not. Then, we have $\Pr[A_{1,2}^{\text{Decide}}] = 1$. Hence, $\text{Adv}^{\text{Medium-Unlin}}(A) > 1/2$. The Lin scheme cannot provide *medium unlinkability*.

In the proposed protocols, i and x are chosen randomly by every individual user. Therefore, no unchanged login message can be derived by the other registered users. Thus, we obtain that $\text{Adv}^{\text{Medium-Unlin}}(A) = 0$. However, each time the user logs in to the server, the server validates his or her identity. So, the advantage $\text{Adv}^{\text{Strong-Unlin}}(A)$ is non-negligible. The proposed schemes cannot achieve *the medium unlinkability*.

6.3.2. Contributory Property of Key Agreement. In the proposed protocols, the session key SK is $h(ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))$. Only for the TAKACP-1 scheme, the server can use the method mentioned in [18, 44] to compute i^* and j^* , thus satisfying $T_i(x) = T_{i^*}(x)$, $T_j(x) = T_{j^*}(x)$, where $T_{i^*}(T_{j^*}(x))$ represents a previous parameter. Since x , $T_i(x)$ and n_1 are randomly selected by the user and SK contains $T_i(x)$ and n_1 , the server still fails to predetermine a session key. Likewise, since $T_j(x)$ and n_2 are randomly selected by the server and SK contains $T_j(x)$ and n_2 , the user cannot predetermine a session key. Notably, neither the server nor the user can determine the specific session key alone in advance. Therefore, the proposed protocols satisfy the contributory property of key agreements.

6.3.3. Session Key Security. Firstly, since i, j, n_1 and n_2 are selected randomly in every run of the protocols, the session key $SK = h(ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))$ is independent of the previously generated session keys. Thus, the proposed protocols can resist against known-key attacks.

Secondly, we demonstrate that the proposed protocols can prevent any inside adversary from computing the session keys. Consider that an inside adversary A_1 has eavesdropped the communication messages $\{e, w_1, \theta_1, w_2, \theta_2, \theta_3\}$ between the user and the server. Since A_1 is a legal user of the same server, A_1 knows n . However, n_1 cannot be derived from e without the server's private keys p and q , where $e = (cn_1)^2 \bmod n$, owing to the quadratic residue assumption. Thus, the adversary cannot still recover $x||T_i(x)$ from w_1 without the knowledge of n_1 , where $w_1 = h(n_1) \oplus (x||T_i(x))$. Moreover, A cannot work out $h(ID||n_1)$. A_1 cannot obtain

$n_2 \| T_j(x)$ from w_2 , since $w_2 = h(\text{ID}^* \| n_1') \oplus (n_2 \| T_j(x))$. Since $\theta_1 = h(c \| x \| T_i(x) \| c_0)$, $\theta_2 = h(c_0^* \| (T_j(x) \oplus h(n_2))) \| \text{SK})$, and $\theta_3 = h(\text{SK}^* \| (n_1 \oplus n_2) \| c_0)$, due to the one-way property of the hash function, A_1 cannot determine $T_i(x)$, $T_j(x)$, or SK . In a word, the session key cannot be derived from the messages transmitted over the public channel. The proposed schemes achieve session key security.

6.3.4. Free Updating of Password. As described in Part C of Section 5, a user can freely update password without any interaction with the server during the password change phase.

6.3.5. Resistance to Password Guessing Attacks. In the proposed TAKACP protocols, the login request message e is information that involves password PW . An inside adversary A_1 may guess the password through the equation $c = d_1 \oplus h(\text{ID} \| PW)$. However, an inside adversary A_1 cannot still obtain c from e since $e = (cn_1)^2 \pmod n$. Therefore, the proposed protocols can resist password guessing attacks.

6.3.6. Resistance to Replay Attacks. The proposed schemes maintain freshness by using two nonces and two chaotic maps. Specifically, the proposed protocols guarantee the freshness of messages by using $T_i(x)$ and n_1 in Step A1, $T_j(x)$ and n_2 in Step A2, and $\{T_i(x), T_j(x), n_1, n_2\}$ in Steps A3 and A4, respectively. Since n_1 is protected by the quadratic residues, only the server and the user itself know it. $T_i(x)$, $T_j(x)$, and n_2 are contained in w_1 and w_2 . They can be calculated only when one knows the nonce n_1 . Therefore, the proposed schemes can prevent replaying attacks.

7. Security, Functionality, and Performance Comparison

In this section, we will make a comparison with the related TAKACP protocols in terms of security, functionality, and performance.

7.1. Security Comparison. We compare the security of our proposed TAKACP schemes with respect to the related authenticated key agreement schemes [19–25, 27–30]. Table 6 summarizes the security properties of the proposed schemes and illustrates the comparison result.

As is indicated in Table 6, the proposed schemes are highly secure as compared to the related authenticated key agreement schemes [19–25, 27–30]. Especially, the proposed schemes can deal with several imperative security issues which most of the authenticated key agreement protocols based on the Chebyshev polynomials defined on the interval $[-1, +1]$ suffer from. For example, the proposed schemes eliminate their weaknesses of the Lin scheme and the Guo-Chang scheme. In contrast, the authentication protocols [22, 23, 25, 28, 29] cannot preserve user anonymity. The protocol in [27] cannot provide the contributory property of key agreement since the session key is determined by the user. The authentication protocols [19, 20, 23] even cannot

provide the session key security. Those protocols presented in [19–21, 23] cannot provide free updating of passwords. The Guo-Chang scheme achieves the anonymity-alone, while the Lin scheme provides weak unlinkability. The proposed schemes achieve the property, medium unlinkability. Note that designing the two-factor authentication protocol with strong unlinkability is still challenging.

7.2. Performance Comparison. In this section, we evaluate the performance of the proposed schemes and make a comparison with the related authenticated key agreement schemes [19–25] in terms of the communication cost, storage, and computational overhead.

We suppose that the block size of secure symmetric cryptosystems is 128 bits, and the output size of a secure one-way hash function is 256 bits. In order to make the factoring problems infeasible in practical implementation, let the module n be an integer of 1024 bits. Since the registration of our schemes is based on a one-way hash function, the password length can be 128 bits. Suppose that the size of ID is 64 bits. In our proposed scheme, the cryptographic parameters $\{c, d\}$ must be stored in the smart card. The length of this information is $1152 + 128 = 1280$ bits, where d can be 128 bits and c must be encrypted in nine blocks. The size of the information stored in the smart card is $64 + 64 + 128 \times 7 + 1024 = 2048$ bits in Fan et al.'s scheme [22], $128 \times 3 + 128 + 64 + 64 + 256 = 896$ bits in Juang et al.'s scheme [23], $128 + 128 \times 3 = 512$ bits in Sun et al.'s scheme [24], $128 \times 3 + 128 + 64 + 64 = 640$ bits in Li et al.'s scheme [25], $128 \times 3 + 256 + 256 = 896$ bits in Guo et al.'s scheme [19], $128 \times 3 + 256 + 128 + 128 = 896$ bits in Lin's scheme [20], and $128 \times 2 + 128 + 128 = 512$ bits in Lee's scheme [21]. As is shown in Table 7, during the registration, the smart card needs a little larger storage space in the proposed schemes than those in other schemes [19–25, 28, 29]. However, it is practically insignificant considering the fact that most current mobile devices, including 4G cellular phones, personal digital assistants (PDAs), and notebook computers, have over a few hundred MB or a few GB of available memory.

In our proposed schemes, the messages transmitted in the registration phase are $\{ID, d\}$. The communication cost of the login protocol is $64 + 256 = 320$ bits. The total size of messages transmitted during the authenticated key exchange phase for cryptographic parameters $\{e, w_1, \theta_1\}$, $\{w_2, \theta_2\}$, and $\{\theta_3\}$ is $(1024 + 256 + 256) + (256 + 256) + 256 = 2294$ bits. The authenticated key exchange phase requires three rounds of message transmission. During the password change, the proposed schemes require no message transmission between the user and the server since the server is not involved with the phase. Let the module number of the elliptic curve be an integer of 163 bits in Juang et al.'s scheme [23] and Sun et al.'s scheme [24]. Let the time stamp be a string of 32 bits in Guo et al.'s scheme [19], Lin's scheme [20], and Lee's scheme [21]. In Juang et al.'s scheme, the communication cost of the login phase about cryptographic parameters $\{b_i, E_{V_i}(e)\}$, $\{u, M_i\}$, and $\{M_U\}$ is $(384 + 384) + (256 + 256) + 256 = 1536$ bits, where $E_{V_i}(e)$ is the encryption of three

TABLE 6: Security comparison.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Fan et al. [22]	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Juang et al. [23]	No	No	Yes	Yes	No	Yes	No	Yes	Yes	No
Sun et al. [24]	No	Anonymity-alone	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Li et al. [25]	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Maitra [27]	No	Medium unlinkability	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Chaudhry et al. [28]	Yes	No	Yes	Yes	Yes	No	#	Yes	Yes	No
Guo et al. [19]	Yes	Anonymity-alone	Yes	Yes	No	Yes	No	Yes	Yes	No
Lin [20]	Yes	Weak unlinkability	Yes	Yes	No	Yes	No	Yes	Yes	No
Lee [21]	Yes	Medium unlinkability	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Irshad et al. [29]	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Irshad et al. [30]	No	Medium unlinkability	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Our protocols	Yes	Medium unlinkability	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

S1: No password table and verification table; S2: User privacy-preserving; S3: Mutual authentication; S4: Contributory property of key agreement; S5: Session key security; S6: Two-factor secrecy; S7: Free updating of password; S8: Resistance to password guessing attacks; S9: Resistance to replay attack. S10: Key confirmation.

TABLE 7: Storage cost and communication cost comparison.

	SC ₁ (in bits)	SC ₂ [*]	SC ₂ (in bits)	SC ₃ [*]	SC ₃ (in bits)	SC ₄ [*]	SC ₄ (in bits)
Fan et al. [22]	2048	3	512	3	1856	#	#
Juang et al. [23]	896	3	578	3	1536	5	2560
Sun et al. [24]	512	2	192	3	1548	0	0
Li et al. [25]	640	1	576	3	3200	5	5248
Maitra [27]	1600	1	1792	2	4416	0	0
Chaudhry et al. [28]	512	1	512	3	2624	#	#
Guo et al. [19]	896	1	320	2	1792	2	1408
Lin [20]	896	1	320	2	1408	2	1280
Lee [21]	512	1	320	2	1088	2	1440
Irshad et al. [29]	768	1	1344	2	1088	0	0
Irshad et al. [30]	1024	1	1344	5	5184	0	0
Our protocols	1280	1	320	3	2294	0	0

SC₁ denotes the storage cost of the smart card in the registration phase. SC₂^{*} denotes the round number of message transmission in the registration phase. SC₂ denotes the total size of the transmitted message in the registration phase. SC₃^{*} denotes the number of message transmissions in the authenticated key exchange phase. SC₃ denotes the total size of the transmitted message in the authenticated key exchange phase. SC₄^{*} denotes the number of the message transmission in the password change phase. SC₄ denotes the total size of transmitted message in the password change phase. # denotes that the scheme does not provide the functionality of password updating.

blocks. The password change phase of Juang et al.'s scheme [23] requires that the user needs to agree on a session key with the server through the log-in phase in advance and then transmit the messages $E_{S_k}(ID_i, h(PW_i^* || b^*))$ and $E_{S_k}(b_i^*)$, respectively. Hence, the size of the message transmitted in the password change phase of Juang et al.'s scheme is 2560 bits. By similar analysis, we can evaluate the communication cost of other related schemes [19–22, 24, 25, 27–30]. The communication cost and storage cost among our schemes and related schemes are shown in Table 7.

As can be seen from Table 7, during the authenticated key exchange phase, the size of the message transmitted between the user and the server in the proposed schemes is a little larger than the size of the message in the schemes [19–24, 29]. However, the proposed schemes can provide the user with stronger privacy protection than the schemes in [22–24]. It also can achieve the session key confirmation, but the schemes [19–21, 29] cannot provide the function. Moreover, the scheme in [29] cannot preserve the anonymity of the user. During the password change phase in the proposed schemes, no message transmission between the user and the server is

required. Compared with the proposed scheme, the server of the other schemes [19–23, 25] is involved with the password change. Furthermore, quite a few messages will be transmitted between the user and the server.

Now, we evaluate the computation cost of our protocols and related protocols. Let T_c denote the time to execute a Chebyshev polynomial computing. Let T_s represent the time to execute a symmetric encryption/decryption operation. We refer to T_h as the time to execute a one-way hash function operation. Let T_m denote the time to execute a scalar multiplication in the elliptic curve group. T_e represents the time to execute one exponentiation operation. We denote by T_{sq} the time to execute a squaring operation. T_{crt} represents the time to solve the square root through the CRT method. Since the XOR operations cost very little, we neglect it. Since a user is required to register with a server one time, the computational cost in the registration phase is not listed in Table 8.

The proposed protocols protect the random number n_1 and the shared secret c by using the quadratic residues. The user requires one modular squaring operation, and the

TABLE 8: Computation overhead comparison.

	C_1	C_2	C_3	C_4
Fan et al. [22]	$4T_h + 1T_{sq} \approx 0.0175$ ms	$3T_h + 1T_s + 1T_{ctt} \approx 0.0238$ ms	#	#
Juang et al. [23]	$3T_h + 3T_s \approx 0.021$ ms	$4T_h + 6T_s + 1T_m \approx 0.186$ ms	$14T_h + 5T_s \approx 0.1015$ ms	$52T_h + 5T_s \approx 0.2345$ ms
Sun et al. [24]	$4T_h + 2T_m \approx 0.232$ ms	$4T_h + 1T_s + 1T_m \approx 0.1335$ ms	$2T_h \approx 0.007$ ms	0
Li et al. [25]	$8T_h + 4T_s \approx 0.0385$ ms	$10T_h + 10T_s + 1T_m \approx 0.249$ ms	$21T_h + 6T_s \approx 0.1365$ ms	$70T_h + 9T_s \approx 0.3395$ ms
Maitra [27]	$8T_h + 6t_e + 1t_m \approx 157.069$ ms	$8T_h + 4t_e + 1t_m + 1t_{inv} \approx 150.029$ ms	$6T_h \approx 0.042$ ms	0
Chaudhry et al. [28]	$1T_p + 6T_h + 3T_m + 2T_a + 2T_e + 2t_m + 1t_{inv} \approx 183.7542$ ms	$3T_p + 5T_h + 4T_m + 3T_a + 2T_e + 1t_m \approx 462.9762$ ms	#	#
Guo et al. [19]	$2T_h + 2T_s + 3T_c \approx 0.3925$ ms	$2T_h + 2T_s + 3T_c \approx 0.3925$ ms	$2T_h + 1T_s + 2T_c \approx 0.2542$ ms	$3T_s + 1T_c \approx 0.153$ ms
Lin [20]	$3T_h + 2T_s + 2T_c \approx 0.2745$ ms	$2T_h + 3T_s + 3T_c \approx 0.403$ ms	$2T_h + 1T_s + 1T_c \approx 0.1327$ ms	$3T_s + 1T_c \approx 0.153$ ms
Lee [21]	$3T_h + 1T_s + 3T_c \approx 0.3945$ ms	$2T_h + 2T_s + 3T_c \approx 0.3925$ ms	$4T_h + 1T_s + 2T_c \approx 0.2675$ ms	$2T_h + 3T_s + 1T_c \approx 0.16$ ms
Irshad et al. [29]	$7T_h + 4T_c \approx 0.5105$ ms	$4T_h + 3T_c \approx 0.3785$ ms	$6T_h \approx 0.042$ ms	0
Irshad et al. [30]	$11T_h + 3T_c \approx 0.4030$ ms	$7T_h + 2T_c \approx 0.2675$ ms	$8T_h \approx 0.056$ ms	0
Our protocols	$9T_h + 1T_{sq} + 1T_c \approx 0.1565$ ms	$7T_h + 1T_{ctt} + 1T_s + 1T_c \approx 0.1593$ ms	$2T_h \approx 0.014$ ms	0

C_1 denotes the computational cost of a user in the authenticated key exchange phase. C_2 refers to the computational cost of the server in the authenticated key exchange phase. C_3/C_4 represents the computational cost of the user/server during the password change phase. N/A represents no requirement of computation. # denotes that the scheme does not provide the functionality of password updating.

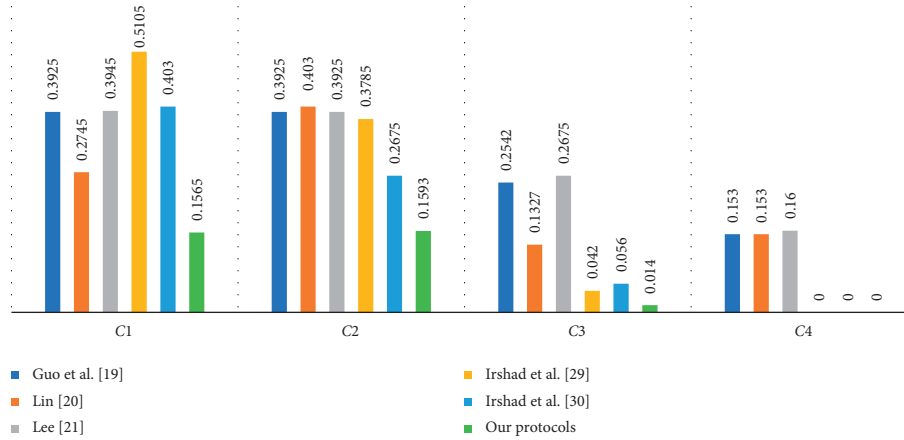


FIGURE 6: Computation overhead comparison of TAKACP protocols (in milliseconds).

server requires one square root solving operation through the CRT and one symmetric decryption. The proposed protocols require no symmetric encryption operation. It needs only one Chebyshev polynomial computing in the user, which is less than one (two) Chebyshev polynomial computing than those in the Lin scheme (the Guo-Chang scheme). The proposed protocol requires one symmetric encryption operation and one Chebyshev polynomial computing in the server, which is less than two (one) symmetric encryption operations and two (two) Chebyshev polynomial computing than those in the Lin scheme (the Guo-Chang scheme). In the proposed protocols, one modular squaring operation does not affect user efficiency since the implementation of one modular squaring [35] can be reduced to only a few hundred gate-equivalents. In practical implementation of the proposed protocols, to efficiently compute the square roots in Z_n^* , the server does the pre-computation [22]. To be specific, S pre-computes and stores the inverse p' of p modular q and the inverse q' of q modular p . In order to compute the square root of a , one first computes $a_1 = a^{(p+1)/4} \bmod p$ and $a_2 = a^{(q+1)/4}$, then he or she can calculate rapidly the four square roots of a in Z_n^* , for example, $(p'a_2 + q'qa_1) \bmod n$. Due to $p \equiv q \equiv 3 \pmod{4}$, the computation of (a_1, a_2) requires about the same time as performing a modular exponentiation computation in Z_n^* . Consequently, we have $1T_{\text{crt}} \approx 1T_e$.

We have executed these operations by utilizing PyCrypto library in Python language in the computer with 16 GB RAM and a clock speed of 3.60 GHz. The time cost of all operations is as follows: $T_h \approx 0.0035$ ms, $T_c \approx 0.1215$ ms, $T_s \approx 0.0105$ ms, $T_m \approx 0.109$ ms, $T_{\text{sq}} \approx 0.0035$ ms, and $T_{\text{crt}} \approx 0.0028$ ms. Table 8 summarizes the computation cost of our scheme with those described in [19–25, 27–30]. As shown in Table 8 and Figure 6, during the authenticated key exchange phase, both the user and the server in the proposed protocols are required at the lowest computation cost among these two-factor authentication protocols [19–21, 29, 30] based on chaotic maps. In addition, the proposed schemes require no involvement of the server during the password change phase. Moreover, in comparison with the related schemes [19–25], the user is required at a very low computation cost during the password change phase of the proposed schemes.

8. Conclusion

In this paper, we examine the limitations of Lin's chaotic map-based authenticated key agreement protocol. We have proposed two TAKACP protocols with key confirmation. Compared with the Lin protocol and the Guo-Chang protocol, the proposed protocols achieve the following additional merits: session key secrecy, medium unlinkability, and free updating of passwords. The proposed protocols with the enhanced security do not affect the user's or the server's efficiency. Therefore, the proposed protocols are highly feasible for practical implementation.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant nos. 61862028 and 61702238), the Natural Science Foundation of Jiangxi Province (Grant no. 20181BAB202016), and the Science and Technology Project of Provincial Education Department of Jiangxi (Grant nos. GJJ160430 and GJJ180288).

References

- [1] Z. Tan, "Secure delegation-based authentication for telecare medicine information systems," *IEEE Access*, vol. 6, no. 1, pp. 26091–26110, 2018.
- [2] W. Ding and W. Ping, "Two birds with one stone: two-factor authentication with security beyond conventional bound," *IEEE Transactions on Dependable and Secure*, vol. 15, no. 4, pp. 708–722, 2018.
- [3] M. Gupta and N. S. Chaudhari, "Anonymous two factor authentication protocol for roaming service in global mobility network with security beyond traditional limit," *Ad Hoc Networks*, vol. 84, no. 1, pp. 56–67, 2019.

- [4] Y. Cao, Q. Zhang, F. Li, S. Yang, and Y. Wang, "PPGPass: nonintrusive and secure mobile two-factor Authentication via wearables," in *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1917–1926, Toronto, Canada, April 2020.
- [5] M. Fotouhi, M. Bayat, A. K. Das, H. A. N. Far, S. Morteza Pournaghi, and M.A. Doostari, "A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care IoT," *Computer Network*, vol. 177, Article ID 107333, 2020.
- [6] M. F. Ayub, S. Shamshad, K. Mahmood, S. H. Islam, R. M. Parizi, and K.-K. R. Choo, "A provably secure two-factor Authentication scheme for USB storage devices," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 4, pp. 396–405, 2020.
- [7] P. Gope and B. Sikdar, "Lightweight and privacy-preserving two-factor authentication scheme for IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580–589, 2019.
- [8] Y. J. Niu and X. Y. Wang, "An anonymous key agreement protocol based on chaotic maps," *Communication Nonlinear Science Numerical Simulators*, vol. 16, pp. 1986–1992, 2011.
- [9] E.-J. Yoon, "Efficiency and security problems of anonymous key agreement protocol based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 7, pp. 2735–2740, 2012.
- [10] Q. Jiang, F. Wei, S. Fu, J. Ma, G. Li, and A. Alelaiwi, "Robust extended chaotic maps-based three-factor authentication scheme preserving biometric template privacy," *Nonlinear Dynamics*, vol. 83, no. 4, pp. 2085–2101, 2016.
- [11] X. Li, J. Niu, S. Kumari et al., "A novel chaotic maps-based user authentication and key agreement protocol for multi-server environments with provable security," *Wireless Personal Communications*, vol. 89, no. 2, pp. 569–597, 2016.
- [12] S. Roy, S. Chatterjee, A. K. Das, S. Chattopadhyay, S. Kumari, and M. Jo, "Chaotic map-based anonymous user authentication scheme with user biometrics and fuzzy extractor for crowdsourcing internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2884–2895, 2018.
- [13] Z. W. Tan, "A privacy-preserving multi-server authenticated key-agreement scheme based on Chebyshev chaotic maps," *Security Communication Networks*, vol. 9, pp. 1384–1397, 2016.
- [14] V. Sureshkumar, R. Amin, M. S. Obaidat, and I. Karthikeyan, "An enhanced mutual authentication and key establishment protocol for TMIS using chaotic map," *Journal of Information Security and Applications*, vol. 53, Article ID 102539, 2020.
- [15] S. Kumari, X. Li, F. Wu, A. K. Das, H. Arshad, and M. K. Khan, "A user friendly mutual authentication and key agreement scheme for wireless sensor networks using chaotic maps," *Future Generation Computer Systems*, vol. 63, pp. 56–75, 2016.
- [16] J. Srinivas, A. K. Das, M. Wazid, and N. Kumar, "Anonymous lightweight chaotic map-based authenticated key agreement protocol for industrial internet of things," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 6, pp. 1133–1146, 2020.
- [17] S. Chatterjee, S. Roy, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 824–839, 2018.
- [18] K. Y. Cheong and T. Koshiba, "More on security of public-key cryptosystems based on chebyshev polynomials," *IEEE Transactions on Circuits & Systems II Express Briefs*, vol. 54, no. 9, pp. 795–799, 2007.
- [19] C. Guo and C.-C. Chang, "Chaotic maps-based password-authenticated key agreement using smart cards," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 6, pp. 1433–1440, 2013.
- [20] H.-Y. Lin, "Improved chaotic maps-based password-authenticated key agreement using smart cards," *Communications in Nonlinear Science and Numerical Simulation*, vol. 20, no. 2, pp. 482–488, 2015.
- [21] T.-F. Lee, C.-H. Hsiao, S.-H. Hwang, and T.-H. Lin, "Enhanced smartcard-based password-authenticated key agreement using extended chaotic maps," *PLoS One*, vol. 12, no. 7, Article ID e0181744, 2017.
- [22] C.-I. Fan, Y.-C. Chan, and Z.-K. Zhang, "Robust remote authentication scheme with smart cards," *Computers & Security*, vol. 24, no. 8, pp. 619–628, 2005.
- [23] W.-S. Juang, S.-T. Chen, and H.-T. Liaw, "Robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 6, pp. 2551–2556, 2008.
- [24] D. Z. Sun, J. P. Huai, J. Z. Sun, J. X. Li, J. W. Zhang, and Z. Y. Feng, "Improvements of Juang 's password-authenticated key agreement scheme using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 2284–2291, 2009.
- [25] X. X. Xiangxue Li, W. D. Weidong Qiu, D. Dong Zheng, K. F. Kefei Chen, and J. H. Jianhua Li, "Anonymity enhancement on robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, pp. 793–800, 2010.
- [26] Q. Xie, D. S. Wong, G. Wang, X. Tan, K. Chen, and L. Fang, "Provably secure dynamic ID-based anonymous two-factor authenticated key exchange protocol with extended security model," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1382–1392, 2017.
- [27] T. Maitra, M. S. Obaidat, R. Amin, S. H. Islam, S. A. Chaudhry, and D. Giri, "A robust ElGamal based password authentication protocol using smart card for client-server communication," *International Journal of Communication Systems*, vol. 30, no. 11, Article ID e3242, 2017.
- [28] S. A. Chaudhry, I. L. Kim, S. Rho, M. S. Farash, and T. Shon, "An improved anonymous authentication scheme for distributed mobile cloud computing services," *Cluster Computing*, vol. 22, no. S1, pp. 1595–1609, 2019.
- [29] A. Irshad, M. Sher, S. A. Chaudhary, H. Naqvi, and M. S. Farash, "An efficient and anonymous multi-server authenticated key agreement based on chaotic map without engaging Registration Centre," *The Journal of Supercomputing*, vol. 72, no. 4, pp. 1623–1644, 2016.
- [30] A. Irshad, S. A. Chaudhry, Q. Xie et al., "An enhanced and provably secure chaotic map-based authenticated key agreement in multi-server architecture," *Arabian Journal for Science and Engineering*, vol. 43, no. 2, pp. 811–828, 2018.
- [31] D. Xiao, X. Liao, and S. Deng, "A novel key agreement protocol based on chaotic maps," *Information Sciences*, vol. 177, no. 4, pp. 1136–1142, 2007.
- [32] M. S. Baptista, "Cryptography with chaos," *Physics Letter A*, vol. 240, no. 1–2, pp. 50–54, 1998.
- [33] L. Kocarev, "Chaos-based cryptography: a brief overview," *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6–21, 2001.

- [34] X. Guo and J. Zhang, "Secure group key agreement protocol based on chaotic hash," *Information Sciences*, vol. 180, no. 20, pp. 4069–4074, 2010.
- [35] T.-F. Lee, "Enhancing the security of password authenticated key agreement protocols based on chaotic maps," *Information Sciences*, vol. 290, pp. 63–71, 2015.
- [36] K. Xue and P. Hong, "Security improvement on an anonymous key agreement protocol based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 7, pp. 2969–2977, 2012.
- [37] Z. Tan, "A chaotic maps-based authenticated key agreement protocol with strong anonymity," *Nonlinear Dynamics*, vol. 72, no. 1-2, pp. 311–320, 2013.
- [38] P. Gong, P. Li, and W. Shi, "A secure chaotic maps-based key agreement protocol without using smart cards," *Nonlinear Dynamics*, vol. 70, no. 4, pp. 2401–2406, 2012.
- [39] X.-Y. Wang and D.-P. Luan, "A secure key agreement protocol based on chaotic maps," *Chinese Physics B*, vol. 22, no. 11, Article ID 110503, 2013.
- [40] H.-Y. Lin, "Chaotic map based mobile dynamic ID authenticated key agreement scheme," *Wireless Personal Communications*, vol. 78, no. 2, pp. 1487–1494, 2014.
- [41] S. H. Islam, M. S. Obaidat, and R. Amin, "An anonymous and provably secure authentication scheme for mobile user," *International Journal of Communication Systems*, vol. 29, no. 9, pp. 1529–1544, 2016.
- [42] S. H. Islam, "Provably secure dynamic identity-based three-factor password authentication scheme using extended chaotic maps," *Nonlinear Dynamics*, vol. 78, no. 3, pp. 2261–2276, 2014.
- [43] X. Hao, J. Wang, Q. Yang, X. Yan, and P. Li, "A chaotic map-based authentication scheme for telecare medicine information systems," *Journal of Medical Systems*, vol. 37, no. 2, pp. 9919–9927, 2013.
- [44] P. Bergamo, P. D'Arco, A. De Santis, and L. Kocarev, "Security of public-key cryptosystems based on Chebyshev polynomials," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 7, pp. 1382–1393, 2005.
- [45] L. Zhang, "Cryptanalysis of the public key encryption based on multiple chaotic systems," *Chaos, Solitons & Fractals*, vol. 37, no. 3, pp. 669–674, 2008.
- [46] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.
- [47] P. Syverson and I. Cervesato, "The logic of authentication protocols," in *FOSAD 2000, LNCS 2171* Springer, Berlin, Heidelberg, 2001.
- [48] K. Bıcakcı and N. Baykal, "One-time passwords: security analysis using BAN logic and integrating with smartcard authentication," *Computer and Information Sciences - ISCIS 2003*, vol. 2869, pp. 794–801, 2003.
- [49] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security (CCS'93)*, pp. 62–73, Fairfax, VA, USA, March 1993.
- [50] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," *Advances in Cryptology - EUROCRYPT 2000*, vol. 18, pp. 139–155, 2000.
- [51] V. Shoup, "Sequences of games: a tool for taming complexity in security proofs," 2005, <http://www.shoup.net>.
- [52] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.