A Lightweight Privacy-Aware Continuous Authentication Protocol-PACA

ABBAS ACAR, Florida International University
SHOUKAT ALI, University of Calgary
KORAY KARABINA, National Research Council Canada
CENGIZ KAYGUSUZ, HIDAYET AKSU, KEMAL AKKAYA, and SELCUK ULUAGAC,
Florida International University

As many vulnerabilities of one-time authentication systems have already been uncovered, there is a growing need and trend to adopt *continuous authentication* systems. Biometrics provides an excellent means for periodic verification of the authenticated users without breaking the continuity of a session. Nevertheless, as attacks to computing systems increase, biometric systems demand more user information in their operations, yielding privacy issues for users in biometric-based continuous authentication systems. However, the current state-of-the-art privacy technologies are not viable or costly for the continuous authentication systems, which require periodic real-time verification. In this article, we introduce a novel, lightweight, privacy-aware, and secure continuous authentication protocol called PACA. PACA is initiated through a password-based key exchange (PAKE) mechanism, and it continuously authenticates users based on their biometrics in a privacy-aware manner. Then, we design an actual continuous user authentication system under the proposed protocol. In this concrete system, we utilize a privacy-aware template matching technique and a wearable-assisted keystroke dynamics-based continuous authentication method. This provides privacy guarantees without relying on any trusted third party while allowing the comparison of noisy user inputs (due to biometric data) and yielding an efficient and lightweight protocol. Finally, we implement our system on an Apple smartwatch and perform experiments with real user data to evaluate the accuracy and resource consumption of our concrete system.

CCS Concepts: • **Security and privacy** → **Authentication**; **Privacy-preserving protocols**; *Biometrics*; *Usability in security and privacy;*

Additional Key Words and Phrases: Continuous, authentication, privacy-aware, biometrics, secure, template, matching

This work was initiated as a collaborative project between Florida International University and Florida Atlantic University, and partially supported by US National Science Foundation (NSF) under the grant numbers NSF-CNS-1718116, 1718109, and NSF-CAREER-CNS-1453647. The statements made herein are solely the responsibility of the authors.
Authors' addresses: A. Acar, C. Kaygusuz, H. Aksu, K. Akkaya, and S. Uluagac, Florida International University, 11200 SW 8th St, Miami, Florida; emails: {aacar001, ckayg001, haksu, kakkaya, suluagac}@fiu.edu; S. Ali, University of Calgary, 2500 University Dr NW, Calgary, Alberta, Canada; email: shoukat.ali@ucalgary.ca; K. Karabina, National Research Council, Canada; email: koray.karabina@nrc-cnrc.gc.ca.

2471-2566/2021/09-ART24 $15.00
https://doi.org/10.1145/3464690

ACM Reference format:
Abbas Acar, Shoukat Ali, Koray Karabina, Cengiz Kaygusuz, Hidayet Aksu, Kemal Akkaya, and Selcuk Uluagac. 2021. A Lightweight Privacy-Aware Continuous Authentication Protocol-PACA. *ACM Trans. Priv. Secur.* 24, 4, Article 24 (September 2021), 28 pages.
https://doi.org/10.1145/3464690

1 INTRODUCTION

Efforts to improve the security of the authentication services have historically progressed from what-you-know (i.e., passwords) to what-you-have (i.e., tokens), then to what-you-are (i.e., biometrics) as attacks have increased in sophistication and become widespread [80, 85]. While the deployment of biometric authentication systems increases the usability of the authentication systems, the plethora of cyber-attacks demands more user information from biometrics, which introduces additional security and privacy challenges in the authentication systems.

In this landscape, another challenge is due to the nature of one-time authentication, which verifies users only at the initial login session regardless of being single- or multi-factor. This is a serious security risk as once the attacker bypasses the initial authentication, it will have a forever access or if the user leaves the system intentionally/unintentionally unlocked, anyone such as an insider or a strong outsider adversary [11], who has physical access to the system will have access to the system without the actual user notification. Therefore, the user should be continuously monitored and re-authenticated. In the literature, several solutions such as time-out or token (or even RFID) based solutions are proposed to address these issues in the authentication systems [55]. Indeed, biometric-based systems are considered to be ideal and usable for such cases as they cannot be easily misplaced unlike tokens, or forgotten unlike passwords, or easily forged by an imposter.

The method of verifying and authorizing the user throughout the session is called *continuous authentication*. A motivational example for continuous authentication would be an enterprise work environment where multiple users can access the same devices. There have already been several attempts to commercialize continuous authentication as a product in enterprise settings [12, 39, 63]. It has also been shown that continuous authentication can be useful to provide extra security when the user is out of a trusted office environment [22] or highly critical/sensitive computing environment such as government offices [64].

Problem Statement. In this study, our goal is ultimately to design a continuous authentication protocol. Since especially behavioral biometrics are ideal in terms of security and usability, this study explores the use of behavioral biometrics for this goal. However, in traditional biometric authentication systems, it is generally assumed that an authentication server and a decision module have access to feature vectors of users in plaintext form [21, 31] during the authentication phase. This, in principle, violates the privacy of biometric data and a malicious/compromised authentication server may infer a lot of useful information about the user. For example, it has been shown that the gender [35, 40], the demographics [30], the emotional state [36], the application context [17] or what the user is typing (e.g., password inference [78, 89]) can be effectively inferred from the keystroke dynamics, which is a behavioral biometric. Therefore, the problem here is to design a privacy-preserving continuous authentication protocol, allowing the computation of the authentication decision on the secure templates generated from the noisy biometric user data. The function to generate the secure templates should not allow to recover the user biometrics from a given template (i.e., irreversibility) and should not allow to link the given template to a user (i.e., unlinkability).

Our Approach. In this article, we tackle these challenges and propose a novel lightweight privacy-aware continuous authentication protocol, called **PACA**. In our protocol, we utilize the

password-authenticated key exchange (PAKE) primitive, which we adapt for the biometric continuous authentication. This provides basic security requirements of our protocol, such as a secure channel between the user and server, mutual authentication, forward secrecy, as well as the resistance against pre-computation attacks. In our design for an actual privacy-aware continuous authentication method, we utilize a secure and noise-tolerant template generation and matching technique called NTT-Sec-$\mathbb{R}$, and combine it with a wearable-assisted continuous authentication method called WACA. NTT-Sec-$\mathbb{R}$ irreversibly transforms the feature vectors, but still allows us to distinguish genuine pairs from imposter pairs. The novel security enhancements proposed in this article are applicable to a wide range of biometric authentication mechanisms when feature vectors are represented as fixed-length real-valued vectors. One of the important applications of such systems is sensor-based keystroke dynamics, which could be used in the authentication of computer [7, 8], smartphone [56], and wearable [38] users.

Contributions. We summarize our contributions as follows:

- We construct a lightweight, privacy-aware, and secure continuous authentication protocol. Previous works have been overlooking this and missing the details of a full protocol. *To the best of our knowledge, our work is the first lightweight privacy-aware protocol* for continuous biometric authentication methods. It is initiated through a PAKE protocol and it continuously authenticates users based on their biometrics. Moreover, it is generic in the sense that one can instantiate it using a large class of secure template generation and matching algorithms, and biometrics-based authentication systems.
- We also design an actual system (the system, its full implementation, and its detailed evaluation) under the proposed protocol: a hybrid (password and keystroke dynamics), continuous, and privacy-preserving biometric authentication for utilized and optimized a wearable-assisted continuous authentication mechanism, and NTT-Sec to handle the real-valued feature vectors while preserving the accuracy. The use of PAKE and NTT-Sec allows one to avoid TLS, any certification authority, verification of certificates, and long-term private keys [41, 75].
- We provide a detailed security and privacy analysis of the proposed protocol against eight different well-known attacks [66] for the biometrics-based authentication methods. We first identify several security requirements. Moreover, we particularly describe detailed attack strategies, and then analyze the resistance of our protocol against those attacks. The security enhancements proposed in this article are applicable to any biometric authentication mechanisms, where feature vectors are represented as fixed-length real-valued vectors.
- We deployed the proposed scheme and provided extensive results with data collected from users wearing an Apple smartwatch to assess the security, accuracy, and resource consumption. More specifically, we provided some concrete estimates for the security of the proposed system, and we report on the timing results, and the false acceptance/rejection rates. Finally, we also measured the resource consumption on a real computing device (e.g., smartwatch).

Organization. The rest of the article is organized as follows: In Section 2, we present our system and security model. Then, in Section 3, we give a detailed description of our privacy-aware biometrics-based continuous authentication protocol. In Section 4, we analyze the basic security requirements of the protocol as well as its robustness against eight different attacks. After that, in Section 5, we present a concrete system under the protocol presented in the previous section and its security analysis. Section 6 presents full implementation details, the accuracy analysis, as well as the resource consumption analysis of our concrete system. Finally, we discuss related work and conclude this article in Sections 7 and 8, respectively.

2 SYSTEM AND SECURITY MODEL

In this section, in order to understand the threat model, we present the basic security requirements of the protocol. Then, we also present the system components and parameters of the protocol, and the assumptions made.

2.1 Security Requirements of the Protocol

The security requirements of our continuous authentication protocol are as follows:

Secure Channel. The communication between the user and the authentication server should be secure against any eavesdropping or interception [13].

Mutual Authentication. The proposed authentication protocol should support the mutual authentication between the user and the authentication server.

Forward Secrecy. This protects past sessions and session keys even after the long-term secret keys of the parties, future sessions, and future sessions keys are compromised.

Resistance against Known Attacks. In addition to the security requirements above, our proposed protocol should be resistant to the main threats known against privacy-aware biometrics-based authentication protocols [62, 66]. We split these threats into four categories: (1) Password recovery attacks, (2) Impersonation attacks, (3) Session intervene attacks, and (4) Biometric recovery attacks. The more details about the attacks and their analysis for PACA are explained in Section 4.

2.2 System Components and Parameters

Our continuous authentication protocol consists of three main components: (1) Password-authenticated Key Exchange (PAKE), (2) User, and (3) Continuous Authentication Server (CAS). Figure 1 illustrates the interactions between the main components of the protocol. (1) User extracts the elements of the feature vector via a feature extraction algorithm and transforms the feature vector to its corresponding template through template generation function. (2,3) After that, the template is transmitted to CAS through the secure channel provided by the PAKE protocol, which also provides the mutual authentication between the user and the server and forward secrecy properties. (4,5) After receiving the user's freshly generated template, the CAS also retrieves the user's already registered template from the database and compares them via the template comparison function. The template comparison function returns a similarity score. The server device decides the authentication result by comparing this similarity score with a predetermined threshold value. In the end, the final authentication result is returned to the user side via the underlying PAKE method. Before explaining the details of our protocol and its components, we also give the description of the symbols used in the protocol in Table 1. In the following subsections, we explain the details of the components.

1. *Password-Authenticated Key Exchange (PAKE).* Our authentication protocol utilizes a (strong) **password-authenticated key exchange (PAKE)** method with some strong security properties. OPAQUE [52] and SRP-6 [87] are examples of such a protocol that satisfy the following features:

(1) **Public Key Infrastructure (PKI)** is not needed because PAKE protocols reduce the security of the system to only the user's password without relying on an outside keying material such as public keys [52]. This is a big advantage from the efficiency point of view because TLS, any certification authority, verification of certificates, long-term private keys, and so on are not required. Another advantage from a security point of view is that any potential failure in PKI is not an issue anymore (such as invalid certificates [1], stolen private keys [2], etc.).
(2) A user and server can mutually authenticate each other.

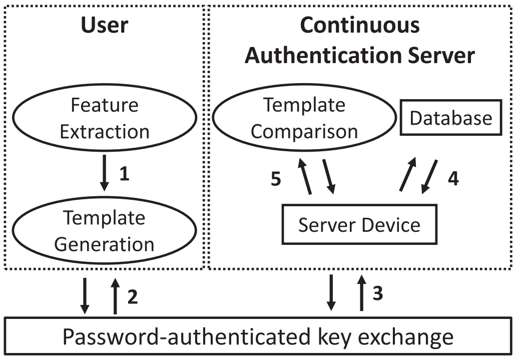


Fig. 1. System components of our proposed continuous authentication protocol, PACA. The detailed definitions are given in Section 2.2.

Table 1. The Symbols Used Throughout This Article

Symbol	Description
User_i	ith user
Id_i	the identity of user i
Pwd_i	ith user's password
$\mathrm{Bio}_{i,t}$	biometrics of the user i at time t
UsrDev_i	ith user's device
$\mathrm{UsrDev}_{i,k}$	the kth device of the ith user
Ext	feature extraction
$\mathrm{f}_{i,t}$	feature vector of user i at time t
$\mathrm{t}_{i,t}$	biometric template of the user i at time t
TempGen	template generation algorithm
TempComp	template comparison algorithm
s	similarity score
DB_i	information of the user i at database
SerDev_k	kth server device
CAS	continuous authentication server
T_i	threshold value of the user i
Time	constant time
NumQuer	number of query
NumMatch	number of match query
MinQuer	constant number of minimum query
MinMatch	constant number of match query
K_i	shared session key between user i and server

(3) The server stores only a cryptographic transformation of the user's password. The password is never sent in clear, and the server does not learn the password of the user.
(4) Pre-computation attacks [52] are not applicable. Such attacks do apply to some password-based protocols if salts are not used, or they are sent in clear from a server to a user, but they do not apply to the specific PAKE instantiations as specified above.
(5) To recover the password of a selected user, the adversary can only mount an exhaustive offline dictionary attack after compromising user data on the server. This attack cannot be avoided but it is computationally not feasible as only the cryptographic transformation of the password is stored on the server.

2. *User*. Throughout this article, $\mathrm{User}_i = (\mathrm{Pwd}_i, \mathrm{Bio}_{i,t}, \mathrm{UsrDev}_i)$ denotes a user indexed with i, her password, her biometric data indexed with t, indicating different measurements of the biometric data of a user, and her device used for collecting the biometric data. A user has access to feature extraction and template generation algorithms:

- $\mathrm{f}_{i,t} \leftarrow \mathrm{Ext}(\mathrm{Bio}_{i,t})$ denotes a feature extraction algorithm. The parameter t considers that different measurements of the same biometric data may result in different feature vectors, i.e., in general, $\mathrm{f}_{i,t_1} \neq \mathrm{f}_{i,t_2}$ for $t_1 \neq t_2$. We assume that the feature extraction always runs on a user device such as user's computer. Biometric data and extracted features are stored only temporarily on this device, and they are deleted after being communicated to another device or entity in our protocol.
- $\mathrm{t}_{i,t} \leftarrow \mathrm{TempGen}(\mathrm{f}_{i,t})$ refers to the one-way transformation of the feature vector into a more secure template, while allowing comparison on the transformed version as well as providing irreversibility and indistinguishability [44] and it corresponds to the traditional hash in password-based systems. Similar to the feature extraction, the operation can be performed on the user device.

3. *Contnuous Authentication Server (CAS)*. CAS denotes an authentication server that validates or invalidates an enrollment or an authentication query initiated by a user (or by an adversary who

is trying to impersonate a user). CAS indicates the validity or invalidity of a query by an output of 1 or 0, respectively. CAS has access to a template comparison algorithm and manages a database and server devices that the users interact with:

- $s \leftarrow \mathsf{TempComp}(\mathsf{t}_{i,t_1}, \mathsf{t}_{i,t_2})$ denotes a template comparison algorithm that takes two templates t_{i,t_1} and t_{j,t_2} captured at two different times as input, and outputs a similarity score, $s \in \mathbb{R}$, quantifying the similarity of the underlying biometric data pair $(\mathsf{Bio}_{i,t_1}, \mathsf{Bio}_{j,t_2})$.
- DB denotes a database that stores information about the users who are enrolled with CAS. For convenience, DB_i denotes the information about User_i. This information is comprised of the user's identity, a cryptographic transformation of her password, and her biometric template along with her matching thresholds. The full definition is given in Section 3.0.1.
- SerDev denotes a server device which the user is trying to log in, such as the desktop or laptop computers of the user. CAS may manage more than one server device. In this case, the kth device of CAS is denoted SerDev_k.

2.3 Assumptions

We list our assumptions regarding the components of the system as follows:

- In general, $\mathsf{Bio}_{i,t_1} \neq \mathsf{Bio}_{i,t_2}$. We assume that each user has at least one device that can extract biometric information of that user. In one of the applications described in this article, we equip users with a smartwatch that extracts the typing behavior of its user. As they are commodity devices, they are easily accessible to many users.
- The user-specific values and devices are distinct and not shared among other users in a regular run of our protocol. More formally, we assume $\mathsf{User}_i \neq \mathsf{User}_j$, $\mathsf{Bio}_{i,t_1} \neq \mathsf{Bio}_{j,t_2}$, and $\mathsf{UsrDev}_i \neq \mathsf{UsrDev}_j$ for $i \neq j$. It also worth noting that a malicious user may control a user device, or a user password may be stolen, but we treat these scenarios as attack scenarios and analyze them in detail to show how our work is robust against these attacks in Section 4.
- A user may have more than one device. In this case, the kth device of the ith user is denoted $\mathsf{UsrDev}_{i,k}$.
- We assume the adversary is a computationally bounded, active adversary who tries to achieve some *adversarial goals* in Section 4.2 to break the security and/or privacy of the users or the system.
- We assume the right user is enrolled during the enrollment phase of the protocol.

3 CONTINUOUS AUTHENTICATION PROTOCOL

In this section, we describe our novel continuous authentication protocol, which includes both the password authentication phase and continuous authentication using the biometrics of the user. Particularly, in our continuous authentication protocol, a user, User_i, is involved in two phases. The enrollment phase is implemented only once and can be implemented at any time before the authentication phase. The authentication phase consists of two parts. The initialization part is implemented only one time, but it has to be implemented every time the user wants to log in. It is required to establish a secure and authentic channel between the user and CAS. Finally, the authentication phase is performed periodically, in which the period depends on the underlying biometrics-based authentication mechanism. We explain the details of enrollment and authentication phases below and illustrate them in Figures 2, 3, and 4, respectively.

3.0.1 Enrollment Phase. In the enrollment phase, a secure template is generated from a biometric trait and stored in CAS. The following are the steps of the enrollment phase:

(1) User_i computes $\mathsf{t}_{i,0} = \mathsf{TempGen}(\mathsf{Ext}(\mathsf{Bio}_{i,0}))$.

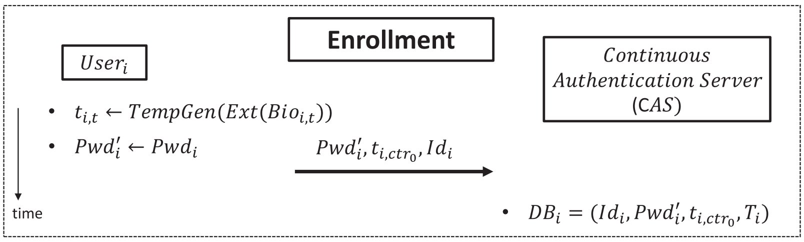

Enrollment
$User_i$
Continuous Authentication Server (CAS)
$t_{i,t} \leftarrow TempGen(Ext(Bio_{i,t}))$
$Pwd'_i \leftarrow Pwd_i$
$Pwd'_i, t_{i,ctr_0}, Id_i$
time
$DB_i = (Id_i, Pwd'_i, t_{i,ctr_0}, T_i)$

Fig. 2. The enrollment phase of our proposed continuous authentication protocol.

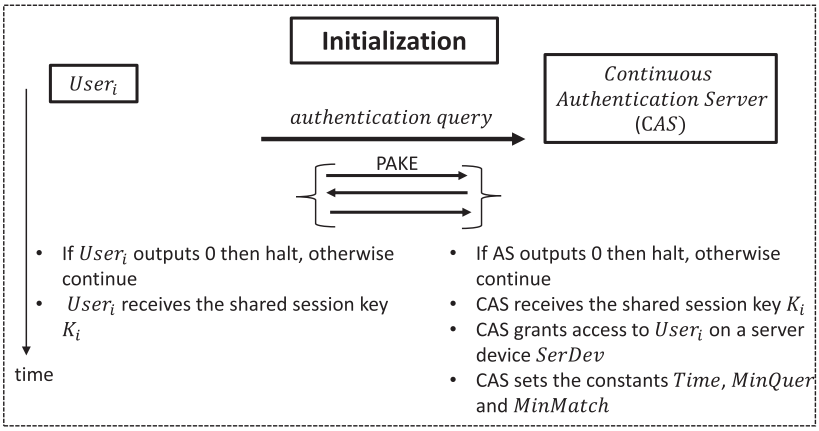

Initialization
$User_i$
Continuous Authentication Server (CAS)
authentication query
PAKE
If $User_i$ outputs 0 then halt, otherwise continue
$User_i$ receives the shared session key K_i
time
If AS outputs 0 then halt, otherwise continue
CAS receives the shared session key K_i
CAS grants access to $User_i$ on a server device $SerDev$
CAS sets the constants $Time$, $MinQuer$ and $MinMatch$

Fig. 3. The initialization phase of our continuous authentication protocol.

(2) User$_i$ registers a cryptographic transformation of her password Pwd$_i$, her template $t_{i,0}$ along with her identity, Id_i by following the underlying PAKE protocol. Note that during enrollment, CAS may want to authenticate User$_i$ and her information through her physical presence. Moreover, CAS may store additional information about the user in DBi such as her matching threshold value T$_i$. The final stored information for each user is shown as $DB_i = (Id_i, Pwd^{'}_i, t_{i,ctr_0}, T_i)$.

3.0.2 Authentication Phase. In the authentication phase, a user's biometric template is periodically verified by the authentication server after a secure and authentic channel is initialized based on PAKE.

Initialization. User$_i$ and CAS execute the underlying PAKE protocol to authenticate each other mutually and generate a session key K_i, which then establishes a secure and authentic channel. If the mutual authentication is successful, then

(1) CAS sets the constants Time, MinQuer, and MinMatch.
(2) CAS initiates a continuous session for User$_i$ (granting access to User$_i$ on the server device, SerDev such as her computer.

Continuous Authentication. In this phase, a user generates her templates and sends them to CAS through the secure and authentic channel established in the initialization phase. Assuming the mutual authentication in the initialization phase is successful, the following steps are executed and are shown in Figure 4:

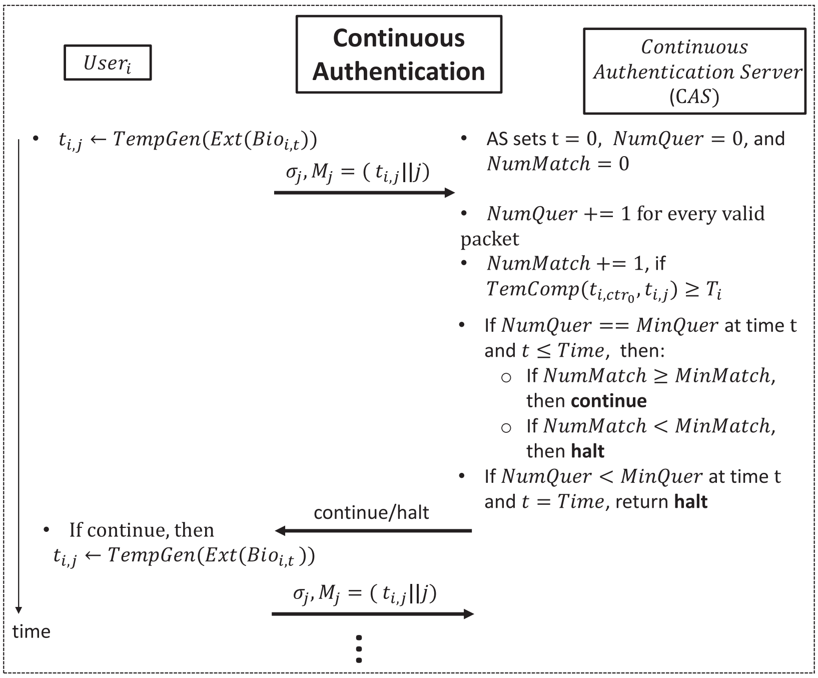

$User_i$
Continuous Authentication
Continuous Authentication Server (CAS)
• $t_{i,j} \leftarrow TempGen(Ext(Bio_{i,t}))$
$\sigma_j, M_j = (t_{i,j}||j)$
• AS sets t = 0, $NumQuer = 0$, and $NumMatch = 0$
• $NumQuer$ += 1 for every valid packet
• $NumMatch$ += 1, if $TemComp(t_{i,ctr_0}, t_{i,j}) \geq T_i$
• If $NumQuer == MinQuer$ at time t and $t \leq Time$, then:
○ If $NumMatch \geq MinMatch$, then **continue**
○ If $NumMatch < MinMatch$, then **halt**
• If $NumQuer < MinQuer$ at time t and $t = Time$, return **halt**
continue/halt
• If continue, then $t_{i,j} \leftarrow TempGen(Ext(Bio_{i,t}))$
$\sigma_j, M_j = (t_{i,j}||j)$
time

Fig. 4. The continuous authentication phase of our continuous authentication protocol.

(1) CAS sets its system time $t = 0$, NumQuer = 0, NumMatch = 0, and keeps track of the system time t.
(2) User$_i$ continuously computes her secure biometric templates $t_{i,j}$ for $j = 1, 2, \ldots,$ and sends packages of the form $(t_{i,j}||j)$[1] to CAS using the confidential and authentic channel established through the shared session key K_i. For each valid package that CAS receives, CAS increments NumQuer by one, and for each $t_{i,j}$ with TempComp$(t_{i,0}, t_{i,j}) \geq T_i$, CAS increments NumMatch by one.
(3) If NumQuer == MinQuer at time t and $t \leq$ Time, then
   - If NumMatch $\geq$ MinMatch, then return (1), indicating the continuity of the session.
   - If NumMatch $<$ MinMatch, then CAS terminates the session and the protocol halts.
(4) If NumQuer $<$ MinQuer in time t and t = Time, then CAS terminates the session and the protocol halts.
(5) If CAS returns 1, the protocol continues from step (2), where the user computes her new biometric template periodically.

If CAS terminates the session, the action to be taken after the termination of the session (e.g., interruption, ended, timed out) depends on the security policy defined by the system administrator. For example, an example policy could be as follows: (1) To return to the initialization phase of the continuous authentication after the first failed attempt, where a new session key is generated between the user and CAS; (2) To request to reset the user password and repeat the enrollment after the three failed attempts phase as it is possible that an intruder is trying to imitate the user.

[1]Here, appending j plays the role of a counter to prevent some obvious replay attacks.

4 SECURITY & PRIVACY ANALYSIS

In this section, we analyze the security requirements of the protocol and we show how PACA is secure and robust against eight well-known malicious attacks against the privacy-aware biometrics-based authentication protocols.

4.1 Analysis of Security Requirements of the Protocol

In our protocol, the basic requirements are provided through the PAKE protocol executed during the initialization phase. A particular example of this PAKE protocol could be OPAQUE [52], which is a strong asymmetric PAKE protocol providing security against pre-computation attacks. In [52], the different versions of the PAKE protocol with different security features are proposed. We specifically consider the version called "the generic OPRF+AKE construction". This version is based on **Oblivious Pseudo-Random Functions (OPRF)** and **Authenticated Key Exchange (AKE)**. The full description of the OPAQUE protocol providing the full forward secrecy and mutual authentication and generating the shared session key between the parties with only three messages exchanged between the user and server can be found in Figure 12 of [52]. In the following subsections, we will show how this specific version of OPAQUE provides the secure channel and satisfies the perfect forward secrecy and mutual authentication in PACA.

Secure Channel. The primary use case of PAKE protocols [52, 87] is that the user does not need to rely on any outside key other than his password. The shared session key K_i is generated from the user's low-entropy password during the execution of the OPAQUE protocol. Then, throughout the entire session, the secure channel is provided through this shared session key. This provides a confidential and authentic communication channel and protects the current session against eavesdropping and man-in-the-middle attacks. Particularly, in PACA, this shared session key K_i is generated during the initialization phase of our protocol after executing the PAKE protocol between the User$_i$ and CAS.

Mutual Authentication. TLS provides only server-side authentication through the certificates, and the password provides the authentication for the user side. However, PAKE protocols achieve mutual authentication without the need for TLS or any PKI infrastructure. For example, the OPAQUE [52] protocol uses HMQV [53] as a base AKE protocol to provide mutual authentication. HMQV extends the computational **Diffie-Hellman (DH)** key exchange with **Exponential Challenge-Response (XCR)** signatures. These signatures are proven to be unforgeable in [53], and they are computed directly on the identity of parties. The ability of parties to provide the signature shows the proof that exchange is carried by the claimed parties and since the messages on which the signatures are computed are directly the identity of the user and server, it proves that the key they computed is uniquely associated with the correct identities (i.e., mutually authenticated).

Forward Secrecy. A protocol is said to have the forward secrecy [53] if the session keys of previous runs cannot be recovered by the attacker after the keys are established, used, and deleted from the memory even after the compromise of long-term keys. Similar to the mutual authentication, the forward secrecy in OPAQUE is provided by the HMQV protocol. The perfect forward secrecy can be achieved if one of the user messages depends on the user's private key. This is achieved by letting DH values by both parties for the session.

4.2 Attack Resistance

In this section, we first present several known attacks against privacy-aware biometrics-based authentication systems [62, 66] and we also analyze our protocol to see if it is robust against these attacks. More specifically, we present the adversarial goals (AGs) by focusing on the adversarial model, attack strategies, their analysis, and the countermeasures our protocol provides against the attacks.

In our proposed continuous authentication protocol, the underlying PAKE method provides the basic security features given in Section 4.1; however, it does not guarantee that the user data will not be revealed to the CAS. This violates the privacy of the user against the server, and to protect the user privacy, in the next section, we propose the use of NTT-Sec-$\mathbb{R}$ for realizing the function $TempGen()$ which is used as a black-box function. The proposed template generation function and its benefits are separately evaluated in Sections 5.3 and 5.4.

AG-1: Password Recovery Attacks. In this attack, the adversary's goal is to recover the password of a user (Pwd_i). The underlying protocol prevents an attacker to read the transmitted packets through the secure channel. An adversary who is capable of actively controlling sessions or compromising a server can achieve this goal only if he succeeds in an exhaustive offline dictionary attack.
Analysis of the attack: Such dictionary attacks cannot be prevented perpetually, but strong passwords would increase the run time of the attack. More precisely, assuming that users choose their passwords uniformly at random from a password space $PassSpace$, then the attack would require $|PassSpace|$ trials. In OPAQUE [52], it has been shown that the cost can be increased by increasing the number of iterations in the hashing operation (i.e., replacing H with H^n in the full protocol). Moreover, we quantify the cost of this attack in Section 5.3.1 particularly for our implementation. Also, note that obtaining the user's password through social engineering techniques is out of this work's scope.

AG-2: Impersonating the User (or the Server) at the Initial Login Phase. In this attack, the adversary's goal is to initiate a session and generate a valid session key (K_i) on behalf of a user, or impersonate a server to a user. These may be achieved by the following two attacks:

(1) *Replay attack:* The adversary may try to replay messages from the previous runs of the protocol between the user and the server.
Analysis of the attack: Such an attempt would fail thanks to the fresh and randomized session keys generated per session with PAKE.
(2) *Password recovery attack through the user impersonation:* If an adversary achieves AG-1 above, she can clearly achieve AG-2. In other words, if the attacker can recover the user's password, then the attacker can impersonate the user during the shared session key generation in the enrollment phase of the authentication.
Analysis of the attack: As in the analysis of AG-1, an offline dictionary attack cannot be prevented perpetually, but strong passwords would increase the run time of the attack.

AG-3: Session Intervene. In this attack, the adversary's goal is to intervene an active session of a user, and to stay undetected as long as possible while behaving maliciously (e.g., interacting with SerDev and impersonating User_i). In the following, we assume that User_i initiates a session with CAS and they both computes the shared session key K_i. We also assume that UsrDev_i stores a copy of K_i. The adversary can achieve AG-3 as follows:

(1) *Package delay attack:* In this first attack scenario, an adversary eavesdrops the communication between User_i and CAS, and interrupts a sequence of legitimate packages (including templates and their counters ($\mathrm{t}_{i,j}||j$), $j = 1, 2, \ldots, k$, encrypted under K_i) going from User_i to CAS. Now, suppose that User_i is out for lunch after sending her last package and leaves the server device SerDev unlocked.[2] Then, the adversary forwards the packages she already collected to CAS while behaving maliciously on SerDev. Receiving sufficiently many

[2]This is a reasonable user behavior in a continuous biometric authentication scheme as such systems assure that adversaries can successfully be detected when they try to impersonate legitimate users.

legitimate packages (e.g., at least MinQuer in time Time), CAS cannot distinguish the adversary from User$_i$, and therefore, the adversary stays undetected and achieves her goal.
Analysis of the attack: This attack can be detected easily if the server device SerDev (i.e., the user's computer) acknowledges CAS immediately after an action is received on SerDev because CAS can detect whether or not the adversary is interrupting and delaying legitimate packages while User$_i$ is legitimately interacting with SerDev.

(2) *Zero-effort and mimicking attacks:* In this second attack scenario, we assume that User$_i$ is out for lunch after initiating a session with CAS and establishing K_i. We also assume that User$_i$ leaves her device UsrDev$_i$ behind, and leaves the server device SerDev unlocked. Now, the adversary captures UsrDev$_i$, and presents her own biometric measurements to CAS (zero-effort attack), or tries to reproduce the physiological or behavioral characteristics of User$_i$ (i.e., imitation attack).
Analysis of the attack: The success rate of this attack would be strongly correlated to the FAR of the system, and the uniqueness of the underlying biometric trait. If the underlying authentication system has a low FAR, the system will identify the real user and the attacker trying to mimic the user, which would prevent the mimicking attacks. See Section 6.1 for the FAR rates of our protocol and see [8] for the robustness of WACA against the mimicking attacks.

(3) *Session key reveal attacks:* The third attack scenario is similar to the second one, but we consider a more powerful adversary. We assume that User$_i$ is not available after initiating a session with CAS, establishing K_i, and sending some packages to CAS (including templates and their counters $(\mathrm{t}_{i,j}|j)$, $j = 1, 2, \ldots, k$, encrypted under K_i). We also assume that User$_i$ leaves her device UsrDev$_i$ behind and the server device SerDev unlocked. In addition, we assume that the adversary recovers the session key K_i from UsrDev$_i$. Having captured some of the previously exchanged packages, the adversary can now recover $\mathrm{t}_{i,j}$ using the knowledge of the key K_i. Next, the adversary can form legitimate packages with the appropriate counters and impersonate User$_i$ during that current session.
Analysis of the attack: In PACA, the communication between CAS and User is protected with the session key, which is created at the beginning of every session using the underlying PAKE protocol. We do not consider this cascaded third attack to be a practical attack because it requires an attacker to extract the session key from the User device in a relatively short amount of time (i.e., before a new session starts, and a new session key is generated).

(4) *Input device replacement attack:* The fourth attack that we consider is a physical attack. After a session is initiated between User$_i$ and CAS, the attacker replaces the legitimate input device of SerDev (e.g., a keyboard or a smartwatch) by her own malicious input device. User$_i$ may still think that she is interacting with SerDev through the legitimate input device, and she may keep sending valid packages to CAS. In the meantime, SerDev receives the adversary's malicious input through the legitimate input device, and CAS keeps the session live based on the legitimate packages it receives from User$_i$.
Analysis of the attack: This attack may work in theory, but it may be challenging to deceive User$_i$ that she is interacting with SerDev through the legitimate input device while indeed she is providing her input through a malicious input device. Therefore, we do not consider this fourth attack to be practical.

AG-4: Recovering Biometrics. In this attack, the adversary tries to recover the biometric information of a user.

(1) *Server compromise attack:* The adversary may be able to capture some of the biometric templates $\mathrm{t}_{i,j}$ of User$_i$ by compromising the server database, or by capturing some of the

packages from a previous session and the session key K_i of that specific session. Then, the adversary can try to reverse the templates ($\mathsf{t}_{i,j}$) back to the biometric information ($\mathsf{Bio}_{i,t}$).
Analysis of the attack: The success rate of the adversary would depend on the difficulty of reversing templates for the given template generation algorithm. Therefore, this attack does not seem to be feasible if an irreversible and indistinguishable template generation algorithm TempGen is deployed. We show the proof of irreversibility and indistinguishability of NTT-Sec-$\mathbb{R}$ in Section 5.3 more formally for our implementation.

4.3 Further Notes on the Attacks, their Limitations, and Justification for Multi-factors

(1) If an adversary captures the secure template of a user $\mathsf{t}_{i,t}$ and a particular session key K_i, but not the password (Pwd_i), then the adversary can impersonate User_i only for that session, because in the next session a fresh session key is generated by the underlying PAKE method and without the knowledge of the new session key, the adversary cannot produce secure templates and legitimate packages to send to CAS.
(2) If an adversary captures the password of a user (Pwd_i), but not the template of a user ($\mathsf{t}_{i,t}$), then she can initiate a session, but her chances for avoiding detection are limited by the success probability of the mimicking attack or the zero-effort attack (or FAR attack).
(3) Another interesting scenario is when an adversary steals the template ($\mathsf{t}_{i,t}$) and the password of a user (Pwd_i). In this case, the adversary can impersonate the user forever unless the user becomes aware and resets the password and re-enrolls in the system. Therefore, one may consider equipping the user device UsrDev with a public-key/private-key pair and involve UsrDev in the session key generation at the beginning of the protocol. For example, a UsrDev signature together with a timestamp can independently be used to confirm that User_i is initiating a session with CAS. In this scenario, the adversary would need the template, password, and also the user device to impersonate User_i. If the user cannot locate her device at any time, she may acknowledge CAS, reset her password, and re-enroll. Moreover, even if the two-way TLS is affordable in the system, a password would still provide an extra barrier for the adversary in case she captures the user's biometric information and device.
(4) So far, we have analyzed the different attack types. Each of these attacks can be instantiated by an outsider or insider. It would be easier for an insider to instantiate a password recovery attack, as an insider would have access to more privilege by having access to the authentication databases. On the other hand, an outsider who is passing through would also instantiate these attacks. However, our analysis above does not change depending on the attacker type in this manner.

We note that, even if UsrDev has a long-term private key and it becomes a part of the protocol in the session key generation, one would still need a biometric factor because otherwise, an adversary would successfully impersonate User_i in lunchtime type attacks by temporarily accessing UsrDev.

The use of a password is also important in our case because PAKE eliminates the need for TLS for mutually authenticating User_i and CAS. Moreover, passwords provide extra protection against an adversary who already captured the user's biometric information and device. In summary, combining all three factors [what you know (password), what you have (device), and who you are (biometrics)] would provide the most comprehensive secure and privacy-aware setup.

5 FULL IMPLEMENTATION

In this section, we describe our hybrid (password and keystroke dynamics), continuous, and privacy-preserving biometric authentication system, which is illustrated in Figure 5. Both for performance evaluation purposes and as a walk-through proof-of-concept case study, we fully

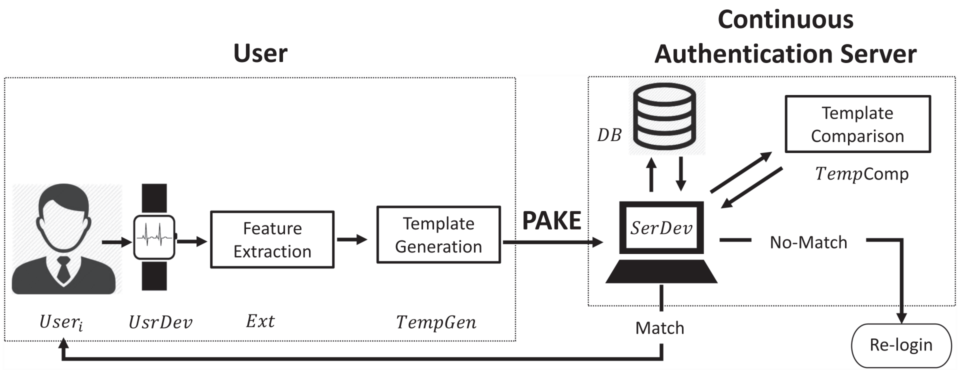


Fig. 5. The architecture of our concrete continuous authentication system used as a case study.

deployed Wearable-Assisted Continuous Authentication framework called WACA [7, 8] as an example continuous biometric authentication system in this study. On the other hand, we utilized NTT-Sec-$\mathbb{R}$ by improving NTT-Sec [49] as a template generation and comparison algorithm, to address the aforementioned privacy issues in continuous biometric authentication settings.

In the following sections, we first explain WACA's design and our modifications. Then, we explain NTT-Sec-$\mathbb{R}$ and show how to use it in the settings of continuous authentication. Finally, we also show how NTT-Sec-$\mathbb{R}$ provides security properties such as irreversibility and indistinguishability while protecting the user templates from the third parties.

5.1 Testing with a Sample Continuous Authentication System

5.1.1 Original WACA. WACA is based on the idea of sensor-based keystroke dynamics, where the authentication data is collected and extracted from the accelerometer and gyroscope sensors of a wearable device (e.g., smartwatch). The raw motion sensor data of the smartwatch is acquired through an app installed on the watch, and the sensor data is transmitted to the computer. In our protocol, the wearable device corresponds to the parameter UsrDev, while the user's computer corresponds to SerDev parameter. The raw accelerometer data is represented in the format of $\vec{acc} = < t_a, x_a, y_a, z_a >$ and gyroscope data is represented as $\vec{gyro} = < t_g, x_g, y_g, z_g >$, where t is timestamp information and x, y, z represent three axis values of the accelerometer and gyroscope sensors. To remove the effect of the noise from data, M-point **Moving Average Filter (MAF)** is applied. Then, for the feature extraction, statistical features (Mean, Median, Variance, Average Absolute Difference of Peaks, Range, Mode, Covariance, **Mean Absolute Deviation (MAD)**, **Interquartile Range (IQR)**, correlation between axes (xy, yz, xz), Skewness, Kurtosis, Entropy, Spectral energy) for the 6-axis data is calculated to obtain a length of 84-point feature vector. Each feature element is calculated over a certain time window of sensor data (i.e., 5 sec, 10 sec). Then, the feature vector is normalized to obtain a scale-invariant vector through the linear normalization. Then, in the authentication phase of WACA, the authentication decision is made by the decision module by computing a similarity score between the feature vector dispatched from the authentication server and the freshly generated feature vector of the current user. In the end, the decision module makes a binary authentication decision (match/no-match) by comparing the similarity score with a predetermined threshold value. If the decision is a no-match, then the user's access to computing terminal is suspended, and the user is required to re-login using the initial authentication method (e.g., password, or 2FA [10, 57]). If the decision is a match, then the user's access is maintained without interrupting the user. This process is repeated periodically with a predetermined period.

5.1.2 New Feature Extraction and Optimization. The WACA dataset consists of 6-axis accelerometer and gyroscope sensor data. The data was collected from the smartwatches of 20 users when the users were typing randomly selected text for 4 minutes (see Typing Task-1 in [8] for more detail). The raw samples are filtered using M-point Moving Average Filter to remove the noise's effect and obtain more smooth sensor data. We obtained the filtered sensor data from the study in [8] and applied the following three optimizations on the original dataset:

- *Feature Extraction:* In the original WACA, statistical features (Mean, Median, Variance, Average Absolute Difference of Peaks, Range, Mode, Covariance, MAD, IQR, the correlation between axes (xy, yz, xz), Skewness, Kurtosis, Entropy, Spectral energy) for 6-axis data is calculated to obtain a length of the 84-point feature vector. Each feature is calculated over the constant sample number (e.g., 1,000 samples). However, we calculated the features over 10 seconds of a constant time interval as it is more intuitive and meaningful. Indeed, it does not affect the results as WACA uses 100-Hz constant frequency during the data collection. At the end of the feature extraction, we obtained between 20-28 samples for each user. For a better analysis, we used 20 feature vectors for each user. In brief, we used 20 feature vectors with 84-features from 20 users in the rest of the article.
- *Evaluation Model:* In the original WACA, one sample from the genuine user is used as an authentication sample, and other samples from the genuine user are used as genuine samples, while all of the samples from other users are used as impostor samples. A score is calculated for each genuine and impostor pair for a predetermined threshold. Then, the rejected samples of the genuine user are used to calculate **False Rejection Ratio (FRR)**, and the accepted samples of the other users' impostor samples are used to calculate **False Acceptance Rate (FAR)**. This is repeated for incrementing the threshold value, and the intersection of the FRR and the FAR plot is reported as EER value. In our paper, we used a user-based evaluation model with training. In particular, we pick the first 10 feature vectors of the ith user for training. Denote this set by $\text{Train}_i = \{[i, j] : j = 1, \ldots, 10\}$, and the remaining 10 feature vectors by $\text{Test}_i = \{[i, j] : j = 11, \ldots, 20\}$. We picked a subset of 5 feature vectors from Train_i, and computed the mean of these 5 feature vectors combinations. This is also called the *gallery* feature vector of a user. As a result, we generated $\binom{10}{5} = 252$ gallery feature vectors per user (simulating 252 different enrollments of a user) and denoted this set by Gallery_i. In our EER_i calculations, we paired each vector from Gallery_i and Test_i. This yielded $252 \cdot 10 = 2520$ genuine comparisons for the ith user. For the ith user, we also paired the first 10 vectors from Gallery_i with all the vectors from Test_j for all $j \neq i$. This yielded $10 \cdot 10 \cdot 19 = 1900$ imposter comparisons for the ith user. We calculated EER for both the original WACA's evaluation model and our optimized evaluation model. We found the original WACA's EER 0.0586 while our optimized evaluation model provided 0.402 EER, where we used 84-point feature vectors for both of them. We consider the reason behind the improved accuracy in our new evaluation model is the training set, which settles down the user's features and results in better accuracy.
- *Feature Selection:* In the original WACA, a feature selection algorithm is not utilized. However, we observed that the 84-point feature vector both affects the security and performance of the system negatively. To prevent this, we improved WACA by applying a feature selection algorithm. Specifically, we applied different univariate feature selection algorithms. The reason we chose univariate algorithms is that we did not want the final feature vector to be dependent on the algorithms used in the decision module. Particularly, we tested three different univariate feature selection algorithms: Chi2 [3], Mutual Information [51], and ANOVA F-test [4]. We re-calculated the EER using the best features selected by these three different

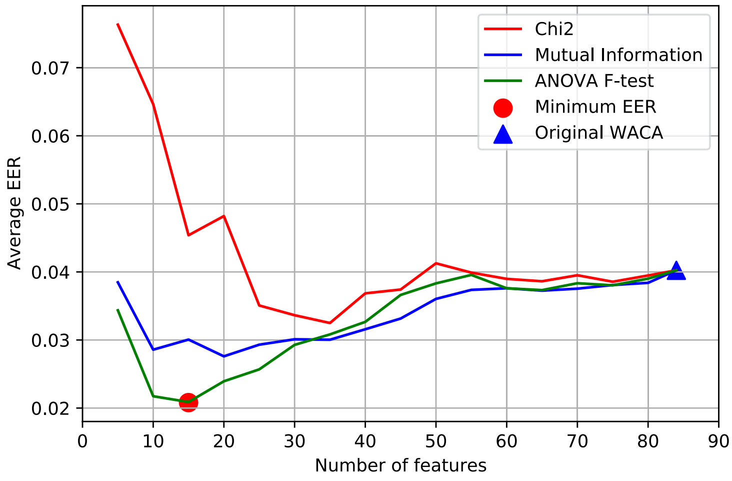


Fig. 6. The graph shows the result of feature selection algorithms. In the graph, ● shows the minimum EER point for all feature selection algorithms, which is obtained using the ANOVA F-test algorithm with 15 feature-point. On the other hand, ▲ shows the 84-feature point, which is the number of features used in the original WACA study.

Table 2. Fifteen Features Chosen by ANOVA F-test Algorithm and Used in Our Experiments

Feature	**Formula**	**ANOVA F-test score**
mean of accelerometer's x-axis	$mean(acc_x)$	1289.51
cross-correlation between accelerometer's x- and z-axis	$sum(abs(xcorr(acc_x, acc_z)))$	989.89
median of accelerometer's x-axis	$median(acc_x)$	626.99
median of accelerometer's y-axis	$median(acc_y)$	497.72
mean of accelerometer's y-axis	$mean(acc_y)$	466.377
entropy of accelerometer's y-axis	$entropy(acc_y)$	377.96
entropy of accelerometer's x-axis	$entropy(acc_x)$	285.51
mean absolute deviation of gyroscope's y-axis	$mad(gyro_y)$	205.32
cross-correlation between accelerometer's y- and z-axis	$sum(abs(xcorr(acc_y, acc_z)))$	175.16
range of gyroscope's y-axis	$range(gyro_y)$	171.11
covariance of gyroscope's y-axis	$cov(gyro_y)$	151.62
spectral energy of gyroscope's y-axis	$sum(fft(gyro_y).*conj(fft(gyro_y)))$	144.86
spectral energy of accelerometer's z-axis	$sum(fft(acc_z).*conj(fft(acc_z)))$	136.50
mean absolute deviation of gyroscope's z-axis	$mad(gyro_z)$	131.15
mean of accelerometer's z-axis	$mean(acc_z)$	122.00

feature selection algorithms. The results are plotted according to a varying number of features in Figure 6. As can be seen in Figure 6, the result of the ANOVA F-test algorithm with the feature vector length of 15 gives the least (average) EER, 0.0208 (shown with ●), while the original WACA's EER was 0.040237 without a feature selection algorithm (shown with ▲). This provided improved 50% performance improvement over the original WACA's accuracy. Moreover, as we will see in Section 5.2, the template generation and comparison function is calculated independently for each feature. Therefore, the fewer the number of features, the less time spent during the template generation and the feature vector's comparison. Therefore, with fewer features, we also obtain improved timing over the original WACA. The top 15 features selected by the ANOVA F-test algorithm are specified in Table 2. For the rest of this article, we use these 15 features of WACA instead of the originally proposed 84 features in WACA.

We applied a feature selection algorithm and selected the top 15 features from WACA. These features included the time and frequency domain statistics (e.g., mean, median, entropy) of the raw sensor values for a certain period. As these features are generic, one could use other biometric authentication methods instead of sensor-based keystroke dynamics, as we did in this article. For example, it could be sensor-based authentication for the smartphone [56], or wearable [38] users.

5.2 Testing with a Secure Template Generation and Comparison Method: NTT-Sec-$\mathbb{R}$

Design Rationale. As mentioned earlier (Section 2.2) that our protocol description requires secure template generation (TempGen) and comparison (TempComp) functions. Our implementation is based on the cryptographic primitive NTT-Sec [49]. We chose NTT-Sec because (1) the NTT-Sec is solely based on publicly computable functions (generalizing cryptographic hash functions in a setting with noisy measurements of data), and (2) NTT-Sec offers formal security analysis with no known attacks to date. Overall, NTT-Sec offers certain advantages over its alternatives. More specifically, (1) homomorphic encryption-based methods [9] are not suitable for the CA protocol due to the requirements of the public key and the private key on the user side; (2) many of the previously known biometric cryptosystems (e.g., fuzzy extractors [47]) are known to have security issues for their reusability [24] and they are limited in their noise tolerance capability; (3) Cancelable biometrics constructions [70], in general, lack formal security analysis, and many constructions have been shown to be vulnerable under false acceptance and stolen key attacks [16].

NTT-Sec consists of two algorithms called Proj (project) and Decomp (decompose). The Proj algorithm maps a length-n binary vector (considered as the feature vector) to a finite field element (considered as its secure template) using an *a priori*-fixed set of public parameters and a factor basis. Given a pair of secure templates, the Decomp algorithm can detect whether the templates originate from a pair of binary feature vectors that differ in at most t indices for an *a priori*-fixed error threshold value t. In Decomp, the detection is achieved by checking whether a particular finite field element can be written (decomposed) as a product of the factor basis elements in a certain form. Computations in NTT-Sec are performed in a cyclotomic subgroup $\mathbb{G}$ of the multiplicative group of a finite field. We adapt the same group structure in our modification. More specifically, let $\mathbb{F}_q$ be a finite field with q elements where $q = p^m$. Let $c \in \mathbb{F}_q$ be a non-quadratic residue with minimal polynomial of degree m over $\mathbb{F}_p$. Let $\mathbb{F}_{q^2} = \mathbb{F}_q(\sigma)$ be a degree-two extension of $\mathbb{F}_q$ where σ is a root of $x^2 - c$. $\mathbb{F}_{q^2}$ has a cyclotomic subgroup $\mathbb{G}$ of order q and every non-identity element in $\mathbb{G}$ can be represented as $\frac{a+\sigma}{a-\sigma}$ for some $a \in \mathbb{F}_q$. Moreover, we say an element $a \in \mathbb{G}$ is *k-decomposable* over $\mathbb{F}_p$ if it can be written as a product $a = \prod_{i=1}^{k}(\frac{a_i+\sigma}{a_i-\sigma})$ for some $\mathbb{F}_p$-elements $a_1, a_2, \ldots, a_k$.

The original NTT-Sec is only limited to working with binary feature vectors by its design. On the other hand, biometric data [7], which we deal with in this article and, in most cases, such as physiological biometrics [72] or behavioral biometrics [38], is represented through real-valued feature vectors. Therefore, we extend NTT-Sec to a new construction, NTT-Sec-$\mathbb{R}$, which comprises two algorithms called NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$. We use the *scale-then-round* transformation in [14] to transform the real-valued feature vectors to integer-valued vectors. Moreover, we describe NTT-Param-$\mathbb{R}$ for the new parameters required in NTT-Sec-$\mathbb{R}$.

5.2.1 NTT-Param-$\mathbb{R}$: System Parameters. We assume that n and t are some fixed values that represent the length of feature vectors and (original) system threshold, respectively. More specifically, a (non-cryptographic) biometric authentication system would declare match for an input pair of biometric data if and only if $d(x, y) \leq t$, where d is the Manhattan distance function (ℓ_1), and x, y are the length-n feature vectors of the biometric data. The parameters of NTT-Sec-$\mathbb{R}$ are defined as follows:

- a scaling factor s;
- a prime number p such that $p > 2n$;
- an integer m such that $m \geq \lfloor st \rceil$;
- a set $\mathrm{B} = \{g_1, g_2, \ldots, g_n\}$ such that $1 \leq g_i \leq \frac{p-1}{2}$ for each i.

We pack all of these parameters under the set $\mathrm{SP} = \{n, t, s, p, m, \mathrm{B}\}$and call this the *system parameter set*. Note that SP can be made public, and commonly used in the NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$ algorithms. SP should be determined in accordance with the desired security parameter λ in order to make NTT-Sec-$\mathbb{R}$ resistant to adversarial attacks.

5.2.2 NTT-Hash-$\mathbb{R}$. NTT-Hash-$\mathbb{R}$ corresponds to the $\mathrm{TempGen}(\mathrm{f}_{i,t})$ function in PACA defined in Section 2.2. Its function is to create a secure template (t_{i,t_1}) from the given user biometric $(\mathrm{f}_{i,t})$. This template should be irreversible and indistinguishable so that the attackers can not infer meaningful information even if they obtain the user templates. This algorithm maps a real-valued feature vector to a $\mathbb{G}$-element called *hash*[3] *value* as follows: Assume a fixed-length real-valued feature vector $x = (x_1, x_2, \ldots, x_n)$ in $[0,1]^n$ is given. Using the *scaling factor* s and the *basis* $\mathrm{B} = \{g_1, g_2, \ldots, g_n\}$, it is mapped to a $\mathbb{G}$-element, defined as

$$\text{NTT-Hash-}\mathbb{R}(x) = \prod_{i=1}^{n} \left(\frac{g_i + \sigma}{g_i - \sigma} \right)^{\lfloor s x_i \rceil}$$

where $\lfloor \cdot \rceil$ is the nearest integer function and the output NTT-Hash-$\mathbb{R}(x)$ is the one-way transformation of the feature vector x.

5.2.3 NTT-Match-$\mathbb{R}$. NTT-Match-$\mathbb{R}$ corresponds to the $\mathrm{TempComp}(\mathrm{t}_{i,t_1}, \mathrm{t}_{i,t_2})$ function in PACA defined in Section 2.2. It is used to match two given templates and decide whether they are generated by the same user or not. For a given hash value $h = \text{NTT-Hash-}\mathbb{R}(x)$ for some $x = (x_1, \ldots, x_n)$ in $[0,1]^n$, a real-valued vector $y = (y_1, \ldots, y_n)$ in $[0,1]^n$ and a positive real number t, the goal of NTT-Match-$\mathbb{R}$ is to decide whether $\sum_{i=1} |x_i - y_i| \leq t$ or not by using their hash values. To achieve this goal, the following process is performed.

NTT-Hash-$\mathbb{R}$ computes $h_y = \text{NTT-Hash-}\mathbb{R}(y)$, and then it decides whether the $\mathbb{G}$-element h/h_y is $\lfloor st \rceil$-decomposable. Furthermore, if the retreived $\mathbb{F}_p$-elements belong to the basis B, NTT-Match-$\mathbb{R}$ returns Match, otherwise No $-$ Match.

5.3 A Security Analysis of NTT-Sec-$\mathbb{R}$

The best strategy for an adversary to attack the new NTT-Sec-$\mathbb{R}$ method (with respect to both irreversibility and indistinguishability notions) is to solve the discrete logarithm problem in the underlying cyclotomic group, which belongs to the finite field $\mathbb{F}_{p^{2m}}$. Discrete logarithms in $\mathbb{F}_{p^{2m}}$ can be computed in time bounded by $(\max(p,m))^{O(\log_2 m)}$ [19]. As analyzed in [49], an attacker needs to solve $(n+1)$ discrete logarithms, and so we calculate the cost of this discrete logarithm attack to be $(n+1)(\max(p,m))^{\log_2 m}$.

5.3.1 Security Levels. In Section 6.1.1, we analyze the security level of our NTT-Sec-$\mathbb{R}$ implementations using the scalars $s = 40$, $s = 100$, and $s = 400$, and denoted by NTTSec_{40}, NTTSec_{100}, and NTTSec_{400}, respectively. The prime number $p = 31$ is chosen for all implementations. Note that the vector length is fixed as $n = 15$. Using these parameters, the security levels λ, which correspond to the minimum cost of the DLP attack [19] and considered as 2^{λ}, are provided in Table 3. In Table 3, each row represents a single user and each user in the system has a different noise profile

[3]We have chosen the name "*hash*"because the algorithm eventually satisfies randomness and irreversibility similar to the hash functions.

Table 3. Security Levels of NTT-Sec-$\mathbb{R}$ for Each user Tested Against the Discrete Logarithm Problem (DLP) Attack [19]

User	DLP					
	NTTSec$_{40}$		NTTSec$_{100}$		NTTSec$_{400}$	
	m	λ	m	λ	m	λ
1	31	29	71	42	281	70
2	41	33	101	48	389	78
3	31	29	73	42	307	72
4	29	28	59	39	227	65
5	43	33	101	48	397	79
6	29	28	67	41	241	67
7	41	33	97	48	367	77
8	53	37	131	53	509	85
9	23	26	59	39	239	66
10	97	48	229	65	919	101
11	37	31	89	46	347	75
12	29	28	67	41	241	67
13	37	31	89	46	337	75
14	67	41	167	59	673	92
15	79	44	191	61	739	95
16	59	39	149	56	587	89
17	19	25	43	33	173	59
18	59	39	137	54	541	86
19	37	31	97	48	359	76
20	47	35	113	51	443	81

Each row represents a single user. We calculated the minimum cost λ (i.e., bit security level) using the formula $\log_2((n+1)(\max(p, m))^{\log_2 m})$ for given the feature length $n = 15$ and prime number $p = 31$ for each user.

in their typing pattern. For example, a user with very consistent typing patterns will introduce less noise/variation in their profile, and will be easier to distinguish from the others . Hence, such users' parameters can be set with smaller m for a fixed scalar s and comparable FAR/FRR. This explains small fluctuations observed in the values of m, whence in their security parameter λ, which is a function of m.

5.3.2 Remark. We note that we are rather conservative in our security analysis, and our estimated bit security levels can be increased in practice at almost no-cost. For example, since a user is already equipped with a password in the protocol, that password can be taken as part of the input in the feature extraction process, while making attacker's task harder in the template reversing attack. This would also allow a legitimate user to revoke his template, and reissue a new template by changing his password and re-enrolling to the system, and also to reuse his biometric data over different systems by choosing different passwords.

5.4 Security Benefits of NTT-Sec-$\mathbb{R}$

In this section, we show the extra security benefits of NTT-Sec-$\mathbb{R}$, in addition to the security properties provided by the PAKE protocol.

- *User Data Privacy.* The user data is very sensitive as it contains the biometrics information so it should be protected from any third party including the authentication server and as well as any kind of eavesdropping. In our protocol, the data is transformed in an indistinguishable and irreversible way before transmitted to any party from the users. Therefore, no party sees the sensitive user data in cleartext. Our system also easily allows the choice of pseudo-identities or user names instead of real user names or identifiers.
- *No Key Required.* The security of NTT-Sec-$\mathbb{R}$ is based on a discrete logarithm problem, where it does not require to store any keys. Therefore, the security of NTT-Sec-$\mathbb{R}$ is not based on a key.

6 PERFORMANCE EVALUATION

In this section, we evaluate our proposed system in terms of accuracy and resource consumption.

Implementation Details. To evaluate the performance of our proposed concrete privacy-aware continuous authentication system, we implemented it on a real system. Specifically, for the timing results of NTT-Sec-$\mathbb{R}$ algorithm's implementation, the codes were written in the C programming language using the GCC 5.4.0 compiler. The Core i7-7700 CPU @ 3.60GHz desktop computer was used with Ubuntu 16.04 LTS running. The CPU time of the match operation was measured using the function clock() from the time.h library. All the timings were provided in milliseconds. For the linear algebra and finite field computations, we used the popular FLINT C-library by William Hart et al. [43]. In all of our implementations, we used the scalars 40, 100, and 400 and denoted by NTTSec_{40}, NTTSec_{100}, and NTTSec_{400}, respectively. Please also note that in all of the experiments in this section, we used the selected 15-point feature vectors calculated after applying ANOVA F-test feature selection algorithm in Section 5.1.1.

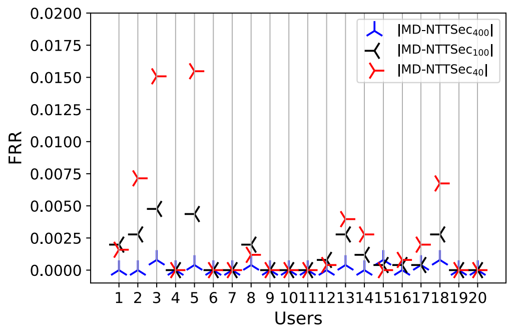
To measure the resource consumption of our proposed continuous authentication system, we used an Apple smartwatch. The results of the resource consumption experiments are given in Figure 8. The feature calculation was strictly done on the device. The measurements were taken on a 38 mm Apple Watch. The feature calculation code was implemented in C. We used KissFFT library [5] for FFT calculations in spectral entropy and cross-correlation features. It is worth noting that we only implemented the most time-consuming parts and the computations on the relatively constrained devices (e.g., smartwatches). The implementation of PAKE was already implemented and analyzed in many works and languages [32, 87] before.

6.1 Accuracy Analysis

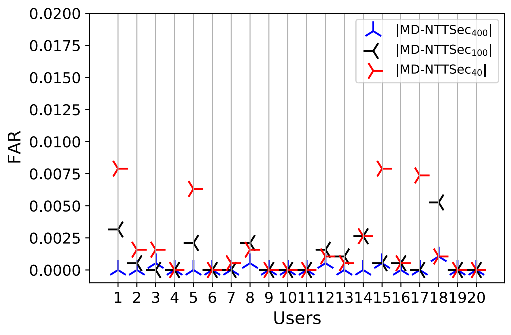
6.1.1 Implementation Results. In this section, we discuss our implementation results. Using the same dataset, we implemented two different techniques: Manhattan Distance (MD) (i.e., no secure template generation) and NTT-Sec-$\mathbb{R}$ algorithm. Unlike the MD, the NTT-Sec-$\mathbb{R}$ algorithm requires the feature vector elements to be an integer; therefore, using the accuracy preserving transformation idea in [14], we transformed real-valued to the integer-valued feature vector. The selection of scalar for scaling purposes is important, and for that reason, we analyzed the experimental results of different (suitable) scalars.

Recall that at the end of our new evaluation model explained in Section 5.1.2, we obtained 1,900 impostor and 2,520 genuine comparison pairs for each user. One distance score for each pair is calculated using Manhattan Distance (MD) resulting with 1,900 impostor and 2,520 genuine distance scores for each user. To decide if a distance score is accepted or not, a threshold needs to be set. For this, we try every threshold value starting from 0 to maximum score and we calculate FRR and FAR for each threshold value. We increment threshold value by 0.001 in each step to obtain accurate FRR and FAR results. We are reporting the FRR and FAR values at the first threshold point where FRR becomes less than or equal to FAR, implying EER. Using the same EER threshold points from the MD results, we determined the parameters for the NTT-Sec-$\mathbb{R}$ algorithm. Among the other parameters, the threshold values were computed as $T = \lfloor s \cdot t \rceil$ where "s" is the (chosen) scalar and "t" is the threshold value obtained from MD results. Moreover, recall that we set the $n = 15$, $p = 31$, and m values are given Table 3. Using these parameters, we compute the secure templates from the feature vectors. Then, we re-calculate the distance scores of 1,900 impostor and 2,520 genuine comparison pairs for scalar value $s = 40$, 100, and 400. Finally, we re-calculate the FRR and FAR values for the EER threshold values from the MD results. To show the change of error rate accuracy, we computed $|\text{MD} - \text{NTTSec}_i|$ using the FRR and FAR values for each user where $i \in \{40, 100, 400\}$. The absolute FRR and FAR differences of the NTTSec implementations w.r.t. the MD are presented in Figures 7(a) and 7(b), respectively.

Figures 7(a) and 7(b) shows the accuracy loss in FRR and FAR, respectively, caused by the NTT-Sec-$\mathbb{R}$ for the scalar values 40, 100, and 400. As shown in both figures, the scalar 40 results in more loss of error rate accuracy than other selected scalars. However, the accuracy loss for both FAR and FRR is less than 0.01 for 18 users out of all 20 users. On the other hand, the accuracy is



(a)



(b)

Fig. 7. (a) The absolute FRR difference between the Manhattan distance (MD) and NTT-Sec-$\mathbb{R}$ implementations using the scalars 40, 100, and 400 for all the 20 users. (b) The absolute FAR difference between the Manhattan distance (MD) and NTT-Sec-$\mathbb{R}$ implementations using the scalars 40, 100, and 400 for all 20 users. Please note that in both figures, the x-axis refers to the single user.

Table 4. The Timing Results of the NTT-Hash-$\mathbb{R}$ Functions of NTTSec$_{40}$, NTTSec$_{100}$, and NTTSec$_{400}$ Algorithms in Milliseconds

User	1	2	3	4	5	6	7	8	9	10
NTTSec$_{40}$	0.373	0.503	0.432	0.386	0.539	0.300	0.5222	0.748	0.241	0.937
NTTSec$_{100}$	1.146	1.586	0.952	0.689	1.602	0.876	1.318	2.212	0.539	3.873
NTTSec$_{400}$	7.258	11.146	7.001	5.124	9.761	3.686	8.798	16.511	4.402	35.544
User	11	12	13	14	15	16	17	18	19	20
NTTSec$_{40}$	0.379	0.284	0.337	1.043	1.123	0.598	0.321	0.438	0.462	0.503
NTTSec$_{100}$	1.189	0.795	1.183	2.588	3.924	2.704	0.631	2.058	1.025	1.395
NTTSec$_{400}$	8.511	3.373	7.507	19.572	27.937	17.536	3.944	16.780	7.601	12.861

We calculate the average time of all users as follows: NTTSec$_{40}$: 0.524 ms, NTTSec$_{100}$: 1.614 ms, and NTTSec$_{400}$: 11.743 ms.

well preserved using the scalar 400. Hence, in terms of accuracy, the higher the scalar is, the lesser the accuracy loss.

6.2 Timing Results

In this section, we report the timing results of NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$ functions. The NTT-Hash-$\mathbb{R}$ function takes the feature vector and finite field parameters as input. Then, the function computes a new field element by using the elements of the feature vector as an exponent of the field element. Therefore, the hash function corresponds to the secure template generation algorithm in PACA (i.e., TempGen). Similarly, the NTT-Match-$\mathbb{R}$ function takes the same finite field parameters and two field elements to perform the comparison. The function outputs Match or No-Match according to the (fixed) threshold. Similar to the hash function, the match function corresponds to the secure template comparison algorithm in PACA (i.e., TempComp). Note that the Match function requires the hashed values of both the query and reference vectors. The reference vectors are stored as hashed values while the query vector is required to be hashed first and then pass to the Match function for comparison. Table 4 and 5 show the timing results of our implementation of NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$ functions, respectively, using the feature vectors we obtained from the WACA dataset for the scalar value 40, 100, and 400.

Table 5. The Average Timing Results of the NTT-Match-$\mathbb{R}$ Functions of NTTSec$_{100}$ and NTTSec$_{400}$ Algorithms in Milliseconds

User	1	2	3	4	5	6	7	8	9	10
NTTSec$_{40}$	0.424	0.726	0.419	0.348	0.758	0.346	0.734	1.240	0.209	4.494
NTTSec$_{100}$	2.667	5.877	2.689	1.302	5.893	2.290	4.530	9.681	1.286	39.652
NTTSec$_{400}$	68.773	154.441	83.268	39.198	143.128	35.313	113.392	290.592	43.850	1408.454
User	11	12	13	14	15	16	17	18	19	20
NTTSec$_{40}$	0.509	0.345	0.511	2.269	3.397	1.301	0.155	1.288	0.514	0.984
NTTSec$_{100}$	4.305	2.264	4.285	15.924	27.268	14.821	0.769	12.034	4.527	6.740
NTTSec$_{400}$	116.286	35.172	97.893	507.976	762.882	378.951	21.117	371.429	109.914	225.091

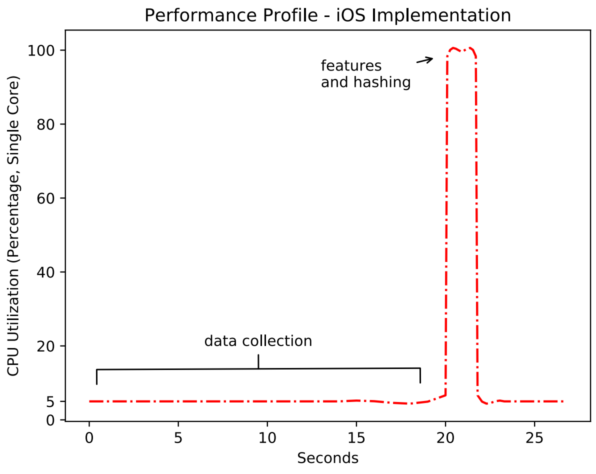
We calculate the average time of all users as follows: NTTSec$_{40}$: 1.483 ms, NTTSec$_{100}$: 8.447 ms, and NTTSec$_{400}$: 256.336 ms.

Table 6. The Comparison of the Timing Performances of Our Proposed Concrete Privacy-aware Continuous Authentication System and Its Alternatives

			Authentication Time (Per Feature)		**Total Authentication**
Ref	**Method**	**Distance Metric**	**User**	**Server**	**Time (User + Server)**
[76] (Simple)	Homomorphic Encryption	Average Absolute Deviation	2326 ms	156 ms	2482 ms
[76] (Optimised)	Order Preserving Encryption		1130 ms	48 ms	1178 ms
[41]	Homomorphic Encryption	Scaled Euclidean Distance	65 ms	≈ 0 ms	65 ms
[41]		Scaled Manhattan Distance	105 ms	6 ms	111 ms
Ours	**NTTSec$_{40}$**	**Manhattan Distance**	**0.03 ms**	**0.07 ms**	**0.10 ms**
Ours	**NTTSec$_{100}$**		**0.11 ms**	**0.56 ms**	**0.67 ms**
Ours	**NTTSec$_{400}$**		**0.78 ms**	**17.09 ms**	**17.87 ms**

In Table 4, the average CPU time of NTT-Hash-$\mathbb{R}$ for all users are given as 0.524 ms, 1.614 ms, and 11.743 for NTTSec$_{40}$, NTTSec$_{100}$, and NTTSec$_{400}$, respectively. On the other hand, the average CPU time of NTT-Match-$\mathbb{R}$ for all users are given as 1.483 ms, 8.447 ms, and 256.336 for NTTSec$_{40}$, NTTSec$_{100}$, and NTTSec$_{400}$, respectively. For these results, we observe that for the same scalar value, NTT-Match-$\mathbb{R}$ takes significantly longer time than NTT-Hash-$\mathbb{R}$, which is similar to the traditional hash functions used for passwords. Moreover, we also see that when the scalar is larger, it takes a longer time to compute both NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$ functions. As we mentioned in Section 5.3.1 and 6.1, increasing the scalar value was increasing the security level significantly while also keeping the accuracy loss limited. However, we also now saw that it was actually also increasing the computation time of the NTT-Hash-$\mathbb{R}$ and NTT-Match-$\mathbb{R}$ functions. Therefore, we end up with the following conclusion: *Smaller scalar means faster computation but more loss of error rate accuracy and less security while larger scalar implies slower computation and lesser loss of accuracy but a more secure system.*

In the literature, there are only limited number of privacy-preserving continuous authentication studies [6, 41, 73, 76, 88]. (Please see Section 7 for more details about these studies.) Among these studies, only two of them [41, 76] report timing performance results. In Table 6, we compare our results with them. To obtain feature-size independent results, we compared the authentication time per feature for all studies. Therefore, for our study, we divide the average timing results obtained from Tables 4 and 5 by 15 and we do the same for [41] too. Moreover, in some of the studies, all of the computation is pushed into either side (user or server); therefore, we also compare the total authentication time. In [76], Shahandashti et al. report two different results, where one of them is called simple and the second one is called optimized as it is assumed that the features can be expressed in 32 bits. Also, in [41], Govindarajan et al. report their results for both Scaled Euclidean and Manhattan distance metrics. We include both of them separately in Table 6.



(a)

(b)

Fig. 8. (a) CPU profile of iOS implementation on an Apple iWatch. The application cruises at 5% on the sensor measurement collection phase. After this, the app calculates the features from sensor data and hashes it, which is illustrated by the peak. The peak's width is approximately 1.6 seconds, which corresponds to the feature extraction (i.e.,Ext) and template generation (i.e.,TempGen) in total. The measurement collection phase follows afterward. This profile repeats until the application is stopped. (b) A screenshot of the application used for the experiments on the Apple iWatch.

The results in [76] show 2, 482 milliseconds total authentication time for the simple design and 1, 178 milliseconds for the optimized design. On the other hand, Govindarajan et al. [41] authors design a privacy-preserving continuous authentication system using homomorphic encryption. They report 65 milliseconds for Scaled Euclidean and 111 milliseconds for Scaled Manhattan distance metrics. Our results of 0.10 milliseconds, 0.67 milliseconds, and 17.87 milliseconds of total authentication time per feature show that our proposed privacy-preserving continuous authentication protocol is significantly lightweight compared to these studies.

6.3 Resource Consumption Analysis

To measure the resource consumption of our proposed continuous authentication system, we implemented our proposed system on an Apple iWatch device. Figure 8 shows how the CPU was utilized throughout the feature calculation period and a screenshot of the application used for the experiments. For 20 seconds, the watch collects data from both the gyroscope and the accelerometer. The device yields 50 measurements per second, giving us 1,000 data points in one dimension. For both gyroscope and accelerometer, we have three dimensions; hence the total count of measurements amount to 6,000. One CPU core is utilized in its complete capacity for a brief amount of time when enough time is elapsed. The total time required for calculations, which is noted in Figure 8 by the peak, is 1.8 seconds; the majority of this time (approximately 90%) is spent to calculate the hash. However, we note that this can be reduced by sending the features to the computer and computing the secure templates on the computer as the computer is also trusted. Therefore, this computation time can be assumed as a lower bound. After, the device begins to collect sensor measurements again, followed by a repeated feature calculation. This profile repeats as long as the framework is in operation.

The memory footprint of the implementation is minuscule and constant, coasting around 3.5 MB. Such a memory profile is expected because upon the completion of the feature calculation period, the previous data is discarded, and since the period and sampling rate is fixed, approximately the same amount of data is recorded anew.

We measured the battery consumption by initiating the algorithm and sampling the overall battery percentage every minute for more than 4 hours. We observed that the application consumed 1% of the battery approximately every 4 minutes, which yields us an operational time of 400 minutes, or 6.5 hours for PACA. Note that this number is an absolute lower bound: during the measurements, a debugger was attached to the device, the device's screen was lit, and the maximum possible values for s and t were selected, greatly increasing computational requirements and the battery consumption. Hence, PACA is very promising in terms of resource consumption.

7 RELATED WORK

Privacy Attacks on Keystroke Dynamics. Even though keystroke dynamics is considered as a good candidate for continuous authentication and identification systems, the information that can leak from the collected keystroke data raises serious privacy concerns. It has been shown that the gender [35, 40], the demographics [30], he emotional state [36] or the application context [17] can be predicted from keystroke dynamics. What the user is typing (e.g., password inference) [78, 89] can also be effectively inferred from the keystroke dynamics. It also has been shown that aggregating the latencies between the consecutive keystrokes does not protect against these attacks [17].

Secure and Privacy-Preserving Biometrics. There are three approaches proposed to address security and privacy issues in biometric schemes: *biometric cryptosystems* (BC), *cancelable biometrics* (CB), and *keyed biometrics* (KB). Some of the key references include [33, 46, 47] for BC, [45, 65, 84] for CB, and [18, 20, 25, 27, 81] for KB. In addition to these three main techniques, there are *hybrid biometrics* (HB), that blend BC, CB, and multi-factor authentication [28, 42]. Some of the above methods have been used to secure multi-biometric traits simultaneously for improved performance and security (see [59] and [61], and the references therein). These constructions can also be considered under HB.

Several theoretical and practical attacks (record-multiplicity, hill-climbing, masquerade attacks, and brute-force attacks) have been developed on BC and CB, many of which result in a total break of the system with respect to irreversibility and indistinguishability. For attacks on BC and CB, see [71], [74], [77], [82], and [86] for BC and [37] and [58] for CB. Several countermeasures have been proposed to guard against these attacks, including hardening with secrets [24, 42, 60], hybrid approaches and multi-biometrics [28, 68], employing encryption or signature schemes [18, 20, 25–27], and new quantization and alignment methods [83]. Recommended safeguards come at the cost of degrading performance and usability, increasing communication and computational bandwidth to impractical ranges, and introducing secret parameters or trusted third parties. These are also the major common problems shared over HB and KB in general. See [59] and [61] for other drawbacks of HB and KB.

Several cryptographic primitives including Secure Multiparty Computation [23, 34], Verifiable Computation [29], and Bloom Filters [67] have been proposed for the secure biometrics. However, the main drawback of the cryptographic primitives is the computational overhead. Moreover, in addition to cryptographic primitives, the biometric template protection methods such as cancellable biometrics [15, 48] and biohashing [69] have been proposed for the secure biometrics. However, Biohashing has been shown as vulnerable to several attacks [50, 54] and even though cancellable biometrics is more secure, they do not apply to behavioral biometrics, which is more ideal for continuous authentication.

We consider an extensive number of attack scenarios and strategies mounted by a powerful adversary in Section 4.2. In particular, the adversary is given the power of capturing biometric templates, session transcripts, and packages; delaying/modifying/replaying packages and sessions while trying to impersonate users with a new session or taking-over a session originally initiated by a genuine user. Our analysis in Section 4.2 supports that our system reasonably resists against all of these attack scenarios with some limitations as explicitly stated in Section 4.3. More specifically, our security analysis with respect to irreversibility and indistinguishability of biometric templates includes brute-force/exhaustive search attacks. We deduce that solving the discrete logarithm problem is the best strategy, and we analyze the security levels of the system with respect to this strategy, and the security estimates are summarized in Table 3 in Section 5.3. Hill-climbing attacks can be prevented during the initial password authentication phase in our system. A more powerful adversary can try to bypass the password authentication phase and try to run hill-climbing during an intervened-session. However, our authentication server responds with a binary value (continue/halt session). In addition, we argue that biometric templates deployed in our system are computationally indistinguishable. Therefore, we expect that the attacker cannot deduce useful information by running hill-climbing type attacks. Similarly, a masquerading attack would try to exploit replaying or reversing templates and/or exploiting FAR/FRR of the system, which we discuss in Section 4.2.

Secure and Privacy-Preserving Continuous Authentication. Although there is extensive literature on the privacy-preserving biometrics, most of the work is on physiological biometrics such as fingerprints, iris, and the like. However, the physiological biometrics are not feasible for a continuous authentication mechanism [79]. A potential solution is to use homomorphic systems [9] to address privacy issues in template matching. A solution using a homomorphic system can be implemented with two main approaches. In the first approach, the user generates a public key-private key pair for HE; the user encrypts his biometric data using his public key, registers it with the server. At the time of verification, the user queries the server with his fresh biometric data encrypted under the same public key. The server uses the public key of the user and computes the encrypted and randomized distance between the template and the queried biometric, and sends it to the user. The user decrypts using his private key and sends the randomized distance back to the server. The server de-randomizes to recover the actual distance and outputs the result (accept or reject). This approach requires users to maintain long-term and individual secret keys, highly interactive with non-trivial computation and bandwidth requirements. See [73] and [76] for some recent implementations of this approach. In the second approach, the server generates public-private key pair for HE, and users encrypt their biometric data under the server's public key during registration and authentication. Even though the key generation/storage/decryption and computations on the encrypted data are performed on two separated independent components of the server, the server has the ability to decrypt and recover the users' biometric data, whence it has to be trusted by all users in the system; see [88] for some recent implementations of this approach. Indeed, it has also been shown that the proposed protocol is vulnerable to biometric template recovery attacks under the presence of even a malicious computational server, which is only one of two servers [6].

To the best of our knowledge, our work is the first lightweight and efficient protocol for privacy-aware continuous biometric authentication methods, which extensively tested for its security and overhead on a real system.

8 CONCLUSION

Unlike the one-time authentication systems, the continuous authentication systems are more suitable and better suited to the contemporary threats in cyberspace. Due to its sensitivity and uniqueness, the biometric data requires proper security and privacy mechanisms in place. Existing solutions like the password-hash-matching systems do not work in noise-tolerant biometric authentication systems, while the privacy-preserving homomorphic encryption constructs are not feasible for continuous authentication due to its performance limitations. In this work, we constructed a lightweight, privacy-aware, and secure continuous authentication protocol and a comprehensive system under the protocol. Formally proving its security and privacy guarantees against eight different attacks, we further deployed our system with NTT-Sec-$\mathbb{R}$ and a continuous biometric authentication system using an Apple smartwatch. We evaluated our protocol's efficiency with data collected from real users and validated that it incurs a minimal overhead. The proposed novel scheme and results can be easily generalized to other biometric authentication mechanisms for both continuous and traditional noncontinuous settings with real-valued feature vectors. Hence, the proposed protocol enables privacy-aware continuous biometric authentication, which can fundamentally improve the security in cyberspace. An interesting future work of our study would be to test PACA with other continuous authentication systems such as touch-based continuous authentication system for smartphone users.

REFERENCES

[1] [n.d.]. https://www.zdnet.com/article/symantec-sacks-staff-for-issuing-unauthorized-google-certificates/. Accessed: 2020-1-20.

[2] [n.d.]. https://www.theverge.com/2019/3/21/18275837/facebook-plain-text-password-storage-hundreds-millions-users. Accessed: 2020-1-20.

[3] [n.d.]. https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html. Accessed: 2020-1-20.

[4] [n.d.]. https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.f_classif.html. Accessed: 2020-1-20.

[5] [n.d.]. KISS FFT. https://github.com/mborgerding/kissfft. Accessed: 2019-07-01.

[6] Aysajan Abidin and Aikaterini Mitrokotsa. 2014. Security aspects of privacy-preserving biometric authentication based on ideal lattices and ring-LWE. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS).* IEEE, 60–65.

[7] Abbas Acar, Hidayet Aksu, Kemal Akkaya, and A. Selcuk Uluagac. 2018. WACA: Wearable-assisted continuous authentication. *IEEE Security and Privacy Workshops (SPW)* (2018).

[8] A. Acar, H. Aksu, A. S. Uluagac, and K. Akkaya. 2020. A usable and robust continuous authentication framework using wearables. *IEEE Transactions on Mobile Computing* (2020), 1–1. https://doi.org/10.1109/TMC.2020.2974941

[9] Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. 2017. A survey on homomorphic encryption schemes: Theory and implementation. *arXiv preprint arXiv:1704.03578* (2017).

[10] Abbas Acar, Wenyi Liu, Raheem Bayeh, Kemal Akkaya, and Arif Selcuk Uluagac. 2019. A privacy-preserving multifactor authentication system. *Security and Privacy* 2, 5 (2019), e88.

[11] Abbas Acar, Long Lu, A. Selcuk Uluagac, and Engin Kirda. 2019. An analysis of malware trends in enterprise networks. In *International Conference on Information Security*. Springer, Cham, 360–380.

[12] Acceptto. [n.d.]. Your Employee and Customer Logins Have Already Been Hacked. https://www.acceptto.com/.

[13] Dharma P. Agrawal and Q. A. Zeng. 2003. Introduction to wireless and mobile systems, brooks. *Cole-Thomson Learning* (2003).

[14] Shoukat Ali, Koray Karabina, and Emrah Karagoz. 2019. Biometric data transformation for cryptographic domains and its application: Poster. In *12th Conference on Security and Privacy in Wireless and Mobile Networks (WiSec 2019), (Miami, FL, May 15-17, 2019).* 304–305. https://doi.org/10.1145/3317549.3326302

[15] Russell Ang, Rei Safavi-Naini, and Luke McAven. 2005. Cancelable key-based fingerprint templates. In *Australasian Conference on Information Security and Privacy*. Springer, 242–252.

[16] Kevin Atighehchi, Loubna Ghammam, Koray Karabina, and Patrick Lacharme. 2020. A cryptanalysis of two cancelable biometric schemes based on index-of-max hashing. *IEEE Transactions on Information Forensics and Security* 15 (2020), 2869–2880.

[17] Kiran S. Balagani, Paolo Gasti, Aaron Elliott, Azriel Richardson, and Mike O'Neal. [n.d.]. The impact of application context on privacy and performance of keystroke authentication systems. *Journal of Computer Security* Preprint ([n. d.]), 1–14.

[18] M. Barbosa, T. Brouard, S. Cauchie, and S. Sousa. 2008. Secure biometric authentication with improved accuracy. *Information Security and Privacy, Lecture Notes in Computer Science* 5107 (2008). Springer, 21–36.

[19] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. 2014. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Advances in Cryptology – EUROCRYPT 2014*, Phong Q. Nguyen and Elisabeth Oswald (Eds.). Springer Berlin, Berlin, 1–16.

[20] M. Barni, T. Bianchi, D. Catalano, M. Raimondo, R. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva. 2010. Privacy-preserving fingercode authentication. In *ACM Workshop on Multimedia and Security* (2010), 231–240.

[21] Debnath Bhattacharyya, Rahul Ranjan, Farkhod Alisherov, and Minkyu Choi. 2009. Biometric authentication: A review. *International Journal of u-and e-Service, Science and Technology* 2, 3 (2009), 13–28.

[22] Blackberry. [n.d.]. *BlackBerry Persona Adaptive Security and AI to Protect Mobile Endpoints.* https://www.blackberry.com/us/en/products/blackberry-persona#industry-focus.

[23] Marina Blanton and Mehrdad Aliasgari. 2009. Secure Computation of Biometric Matching. Department of Computer Science and Engineering, University of Notre Dame, Tech. Rep. 3 (2009), 2009.

[24] M. Blanton and M. Aliasgari. 2011. On the (non-)reusability of fuzzy sketches and extractors and security in the computational setting. In *International Conference on Security and Cryptography (SECRYPT'11)* (2011).

[25] M. Blanton and P. Gasti. 2011. Secure and efficient protocols for iris and fingerprint identification. In *, European Symposium on Research in Computer Security (ESORICS'11).* (2011), 190–209.

[26] X. Boyen. 2004. Reusable cryptographic fuzzy extractors. In *11th ACM Conference on Computer and Communications Security* (2004), 82–91.

[27] J. Bringer, H. Chabanne, M. Izabachène, D. Pointcheval, Q. Tang, and S. Zimmer. 2007. An application of the Goldwasser-Micali cryptosystem to biometric authentication. In *Information Security and Privacy, Lecture Notes in Computer Science,* vol. 4586 (2007), Springer, 96–106.

[28] J. Bringer, H. Chabanne, and B. Kindarji. 2008. The best of both worlds: Applying secure sketches to cancelable biometrics. *Science of Computer Programming* 74 (2008), 43–51.

[29] Julien Bringer, Hervé Chabanne, Firas Kraïem, Roch Lescuyer, and Eduardo Soria-Vázquez. 2015. Some applications of verifiable computation to biometric verification. In *2015 IEEE International Workshop on Information Forensics and Security (WIFS).* IEEE, 1–6.

[30] David Guy Brizan, Adam Goodkind, Patrick Koch, Kiran Balagani, Vir V. Phoha, and Andrew Rosenberg. 2015. Utilizing linguistically enhanced keystroke dynamics to predict typist cognition and demographics. *International Journal of Human-Computer Studies* 82 (2015), 57–68.

[31] Paul M. Burger. 2001. Biometric authentication system. US Patent 6,219,439.

[32] Tom Cocagne. [n.d.]. Minimal C implementation of the Secure Remote Password protocol (version 6a). https://github.com/cocagne/csrp.

[33] Y. Dodis, L. Reyzin, and A. Smith. 2004. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptology (Eurocrypt 2004). Lecture Notes in Computer Science* vol. 3027 (2004), Springer, 523–540. Updated by Y. Dodis, L. Reyzin, and A. Smith in 2008.

[34] David Evans, Yan Huang, Jonathan Katz, and Lior Malka. 2011. Efficient privacy-preserving biometric identification. In *17th Conference Network and Distributed System Security Symposium (NDSS).*

[35] Michael Fairhurst and Márjory Da Costa-Abreu. 2011. Using keystroke dynamics for gender identification in social network environment. In *4th International Conference on Imaging for Crime Detection and Prevention (ICDP'11).* IET, 1–6.

[36] Michael Fairhurst, Cheng Li, and Meryem Erbilek. 2014. Exploiting biometric measurements for prediction of emotional state: A preliminary study for healthcare applications using keystroke analysis. In *2014 IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications (BIOMS).* IEEE, 74–79.

[37] Y. Feng, M-H. Lim, and P. Yuen. 2014. Masquerade attack on transform-based binary-template protection based on perceptron learning. *Pattern Recognition* 47 (2014), 3019–3033.

[38] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. 2013. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Transactions on Information Forensics and Security* 8, 1 (2013), 136–148.

[39] Gatekeeper. [n.d.]. Continuous Authentication. https://gkaccess.com/support/glossary/continuous-authentication/.

[40] Romain Giot and Christophe Rosenberger. 2012. A new soft biometric approach for keystroke dynamics based on gender recognition. *International Journal of Information Technology and Management* 11, 1–2 (2012), 35–49.

[41] Sathya Govindarajan, Paolo Gasti, and Kiran S. Balagani. 2013. Secure privacy-preserving protocols for outsourcing continuous authentication of smartphone users with touch data. In *2013 IEEE 6th International Conference on Biometrics: Theory, Applications and Systems (BTAS).* IEEE, 1–8.

[42] F. Hao. 2006. Combining crypto with biometrics effectively. *IEEE Trans. Comput.* 55 (2006), 1081–1088.

[43] W. Hart, F. Johansson, and S. Pancratz. 2013. FLINT: Fast Library for Number Theory. Version 2.4.0, http://flintlib.org.

[44] Anil K. Jain, Karthik Nandakumar, and Abhishek Nagar. 2008. Biometric template security. *EURASIP Journal on Advances inSignal Processing 2008* (2008), 113.

[45] A. Jin, D. Ling, and A. Goh. 2004. Biohashing: Two factor authentication featuring fingerprint data and tokenised random number. *Pattern Recognition* 37 (2004), 2245–2255.

[46] A. Juels and M. Sudan. 2006. A fuzzy vault scheme. *Designs, Codes and Cryptography* 38 (2006), 237–257.

[47] A. Juels and M. Wattenberg. 1999. A fuzzy commitment scheme. In *6th ACM Conference on Computer and Communications Security* (1999), 28–36.

[48] Sanjay Kanade, Dijana Petrovska-Delacrétaz, and Bernadette Dorizzi. 2009. Cancelable iris biometrics and using error correcting codes to reduce variability in biometric data. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009 (CVPR 2009)*. IEEE, 120–127.

[49] Koray Karabina and Onur Canpolat. 2016. A new cryptographic primitive for noise tolerant template security. *Pattern Recognition Letters* 80 (2016), 70–75.

[50] Adams Kong, King-Hong Cheung, David Zhang, Mohamed Kamel, and Jane You. 2006. An analysis of BioHashing and its variants. *Pattern Recognition* 39, 7 (2006), 1359–1368.

[51] L. F. Kozachenko and Nikolai N. Leonenko. 1987. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii* 23, 2 (1987), 9–16.

[52] H. Krawczyk. [n.d.]. The OPAQUE Asymmetric PAKE Protocol draft-krawczyk-cfrg-opaque-00. https://tools.ietf.org/html/draft-krawczyk-cfrg-opaque-00.

[53] Hugo Krawczyk. 2005. HMQV: A high-performance secure Diffie-Hellman protocol. In *Annual International Cryptology Conference*. Springer, 546–566.

[54] Karl Kümmel and Claus Vielhauer. 2010. Reverse-engineer methods on a biometric hash algorithm for dynamic handwriting. In *12th ACM Workshop on Multimedia and Security*. ACM, 67–72.

[55] Stan Kurkovsky, Ewa Syta, and Bernardo Casano. 2010. Continuous RFID-enabled authentication and its privacy implications. In *2010 IEEE International Symposium on Technology and Society*. IEEE, 103–110.

[56] Wei-Han Lee and Ruby B. Lee. 2017. Sensor-based implicit authentication of smartphone users. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 309–320.

[57] Wenyi Liu, A. Selcuk Uluagac, and Raheem Beyah. 2014. MACA: A privacy-preserving multi-factor cloud authentication system utilizing big data. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 518–523.

[58] A. Nagar, K. Nandakumar, and A. Jain. 2010. Biometric template transformation: A security analysis. In *Proceedings of SPIE, Electronic Imaging, Media Forensics and Security II* 7541 (2010), 1–15. Invited paper.

[59] K. Nandakumar and A. Jain. 2015. Biometric template protection: Bridging the performance gap between theory and practice. *IEEE Signal Processing Magazine* 32 (2015), 88–100.

[60] K. Nandakumar, A. Nagar, and A. Jain. 2007. Hardening fingerprint fuzzy vault using password. In *Advances in Biometrics, Lecture Notes in Computer Science*, vol. 4642 (2007), 927–937.

[61] I. Natgunanathan, A. Mehmood, Y. Xiang, G. Beliakov, and J. Yearwood. 2016. Protection of privacy in biometric data. *IEEE Access* 4 (2016), 880–892.

[62] Elena Pagnin and Aikaterini Mitrokotsa. 2017. Privacy-preserving biometric authentication: Challenges and directions. *Security and Communication Networks* 2017 (2017).

[63] Plurilock. [n.d.]. GLOSSARY TERM Continuous Authentication. https://www.plurilock.com/products/adapt/.

[64] Plurilock. [n.d.]. Plurilock Successfully Meets Second Milestone in Department of Homeland Security Contract. https://www.plurilock.com/press-release/plurilock-successfully-meets-second-milestone-in-department-of-homeland-security-contract/.

[65] N. Ratha, S. Chikkerur, J. Connell, and R. Bolle. 2007. Generating cancelable fingerprint templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29 (2007), 561–572.

[66] Nalini K. Ratha, Jonathan H. Connell, and Ruud M. Bolle. 2001. Enhancing security and privacy in biometrics-based authentication systems. *IBM Systems Journal* 40, 3 (2001), 614–634.

[67] Christian Rathgeb, Frank Breitinger, Christoph Busch, and Harald Baier. 2014. On application of Bloom filters to iris biometrics. *IET Biometrics* 3, 4 (2014), 207–218.

[68] C. Rathgeb, B. Tams, J. Wagner, and C. Busch. 2016. Unlinkable improved multi-biometric iris fuzzy vault. *EURASIP Journal on Information Security* 26 (2016), 1–16.

[69] Christian Rathgeb and Andreas Uhl. 2010. Iris-biometric hash generation for biometric database indexing. In *2010 20th International Conference on Pattern Recognition (ICPR)*. IEEE, 2848–2851.

[70] Christian Rathgeb and Andreas Uhl. 2011. A survey on biometric cryptosystems and cancelable biometrics. *EURASIP Journal on Information Security* 2011, 1 (2011), 3.

[71] C. Rathgeb and A. Uhl. 2012. Statistical attack against fuzzy commitment scheme. *IET Biometrics* 1 (2012), 94–104.

[72] Ajita Rattani, Dakshina Ranjan Kisku, Manuele Bicego, and Massimo Tistarelli. 2007. Feature level fusion of face and fingerprint biometrics. In *2007 1st IEEE International Conference on Biometrics: Theory, Applications, and Systems*. IEEE, 1–6.

[73] Nashad Ahmed Safa, Reihaneh Safavi-Naini, and Siamak F. Shahandashti. 2014. Privacy-preserving implicit authentication. In *IFIP International Information Security Conference*. Springer, 471–484.

[74] W. Scheirer and T. Boult. 2007. Cracking fuzzy vaults and biometric encryption. In *Biometrics Symposium* (2007), 1–6.

[75] Jaroslav Šeděnka, Sathya Govindarajan, Paolo Gasti, and Kiran S. Balagani. 2015. Secure outsourced biometric authentication with performance evaluation on smartphones. *IEEE Transactions on Information Forensics and Security* 10, 2 (2015), 384–396.

[76] Siamak F. Shahandashti, Reihaneh Safavi-Naini, and Nashad Ahmed Safa. 2015. Reconciling user privacy and implicit authentication for mobile devices. *Computers & Security* 53 (2015), 215–233.

[77] K. Simoens, P. Tuyls, and B. Preneel. 2009. Privacy weaknesses in biometric sketches. In *2009 30th IEEE Symposium on Security and Privacy* (2009), 188–203.

[78] Dawn Xiaodong Song, David Wagner, and Xuqing Tian. 2001. Timing analysis of keystrokes and timing attacks on SSH. In *Proceeding of the 10th USENIX Security Symposium*. USENIX Association, Berkley, CA, 337–352.

[79] Riccardo Spolaor, QianQian Li, Merylin Monaro, Mauro Conti, Luciano Gamberini, and Giuseppe Sartori. 2016. Biometric authentication methods on smartphones: A survey. *PsychNology Journal* 14, 2 (2016).

[80] Deian Stefan, Xiaokui Shu, and Danfeng Daphne Yao. 2012. Robustness of keystroke-dynamics based biometrics against synthetic forgeries. *Computers & Security* 31, 1 (2012), 109–121.

[81] A. Stoianov. 2010. Cryptographically secure biometric. *SPIE, Biometric Technology for Human Identification VII* 7667 (2010). 12 pages.

[82] B. Tams. 2014. Decodability attack against the fuzzy commitment scheme with public feature transforms. arxiv.org/abs/1406.1154.pdf.

[83] B. Tams, P. Mihailescu, and A. Munk. 2015. Security considerations in minutiae-based fuzzy vaults. *IEEE Transactions on Information Forensics and Security* 10 (2015), 985–998.

[84] A. Teoh, A. Goh, and D. Ngo. 2006. Random multispace quantization as an analytic mechanism for BioHashing of biometric and random identity inputs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), 1892–1901.

[85] Chee Meng Tey, Payas Gupta, and Debin Gao. 2013. I can be you: Questioning the use of keystroke dynamics as biometrics. In *Proceedings o the Annual Network and Distributed System Security Symposium 20th NDSS 2013, 24-27 February*. 1–16.

[86] Y. Wang, S. Rane, S. Draper, and P. Ishwar. 2012. A theoretical analysis of authentication, privacy, and reusability across secure biometric systems. *IEEE Transactions on Information Forensics and Security* 7 (2012), 1825–1840.

[87] Tom Wu. [n.d.]. The Stanford SRP Homepage. http://srp.stanford.edu/.

[88] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. 2013. Packed homomorphic encryption based on ideal lattices and its application to biometrics. In *International Conference on Availability, Reliability, and Security*. Springer, 55–74.

[89] Kehuan Zhang and XiaoFeng Wang. 2009. Peeping tom in the neighborhood: Keystroke eavesdropping on multi-user systems. In *Proceedings of the USENIX Security Symposium*. 17–32.

Received March 2020; revised March 2021; accepted May 2021