Collaborative Authentication using Threshold Cryptography

Aysajan Abidin [1], Abdelrahaman Aly [1], and Mustafa A. Mustafa [1,2]

[1] KU Leuven, imec-COSIC, Belgium
[2] The University of Manchester, UK
firstname.lastname@esat.kuleuven.be

Abstract. We propose a collaborative authentication protocol where multiple user devices (e.g., a smartphone, a smartwatch and a wristband) collaborate to authenticate the user to a third party service provider. Our protocol uses a threshold signature scheme as the main building block. The use of threshold signatures minimises the security threats in that the user devices only store shares of the signing key (i.e., the private key) and the private key is never reconstructed. For user devices that do not have secure storage capability (e.g., some wearables), we propose to use fuzzy extractors to generate their secret shares using behaviometric information when needed, so that there is no need for them to store any secret material. We discuss how to reshare the private key without reconstructing it in case a new device is added and how to repair shares that are lost due to device loss or damage. Our implementation results demonstrate the feasibility of the protocol.

Key words: Collaborative authentication, security and privacy, threshold signatures, secret sharing, fuzzy extractors.

1 Introduction

The increasing number of mobile and wearable devices being carried by users results in more sensitive information being stored on and/or accessed via these devices. This provides users with more flexibility in terms of accessing resources and services, thus enhancing their personal experience and convenience, as well as, it creates new opportunities for both users and service providers. However, this flexibility comes at a cost, introducing new security and privacy challenges [1]. For example, some of these personal devices, i.e., wearables (such as smartwatches and wristbands), unlike smartphones, have limited computational and interaction capabilities, thus making them unsuitable to use existing authentication protocols. The use of easily-accessible context information such as the user's location and typical behaviour causes privacy concerns too. In addition, since these devices are small, light, and easy to carry, they are also prone to loss,

theft and/or break. Nevertheless, users still expect strong security, privacy protection as well as maximum flexibility and convenience when accessing services or resources provided by the service providers. Satisfying the users' needs while minimising the associated security and privacy risks of using personal and wearable devices will require new, more collaborative, ways for users to be authenticated and granted access to a wide range of on-line services and content [2].

Existing solutions for user authentication [**?**, 3–7] do not satisfy these needs: (i) users prefer password-less solutions, (ii) biometric-only solutions have security and usability tradeoff, and (iii) wearable devices have limited computational powers and are more prone to loss and theft. Thus, authentication solutions that are more user-, device- and context-tailored are necessary.

One way to achieve this is to use a collaborative approach in designing authentication schemes. Collaborative authentication schemes are the ones where multiple devices jointly authenticate to a remote server or within a device-to-device setting with minimum user effort. They are getting traction as users carry multiple devices and wearables with themselves nowadays. To limit the cost, the combination of wearables and the user's other devices, say, a smartphone, would be preferred. Such collaborative authentication schemes overcome the security problems of using a single possession factor or a knowledge factor during the authentication process. An adversary would have to compromise multiple wearables to successfully impersonate a user. Moreover, by using wearables the user is carrying anyhow, one avoids the need of employing external hardware authentication tokens, which could be quite costly.

The main concept of collaborative authentication is to transform a challenge-response protocol with a single prover and verifier, to a challenge-response protocol with multiple collaborating provers and a single verifier. To mitigate the threat of wearables being stolen or lost, and the fact that the set of wearables is dynamic (the user is not always carrying the same set of wearables), threshold-based cryptography can be used. The aim of threshold cryptography is to protect a key by sharing it amongst a number of entities in such a way that only a subset of minimal size, namely a threshold t (out of, say, $n > t$), can use the key. No information about the key can be learnt from $t - 1$ or less shares. To deal with user devices that do not have secure element to store their share, such devices can make use of fuzzy extractors to generate their shares of the secret key from users' biometric data on demand whenever they need them.

Contribution. The main contributions of this paper are summarised as follows.

- Firstly, it proposes a concrete collaborative authentication protocol that uses threshold signatures.
- Secondly, it presents mechanisms for key reshare and share repair in case some of the original shares of the key are lost or compromised. For repairing lost shares, we use a repairable threshold scheme proposed in a recent paper by Stinson and Wei in [8].
- Thirdly, it proposes a secret share generation mechanism for devices with no memory applying fuzzy extractors to behaviometric information captured by the device sensors, improving the usability of the authentication protocol. We do note that the same procedure also applies to PUFs (physically uncloneable functions) implemented in the devices.
- Lastly, it presents performance results from the implementation of the share regeneration and threshold signatures. The results show that the proposed protocol is feasible in practice.

Outline. The remainder of this paper is organised as follows: Section 2 presents our system model and requirements for collaborative authentication schemes. Section 3 presents security definitions of the cryptographic building blocks and our threat model. Then, Section 4 proposes a concrete collaborative authentication protocol that combines threshold signatures and fuzzy extractors, and introduces a share generation mechanism as well as the threshold Schnorr signature scheme using secret sharing. Sections 5 and 6 present the security analysis and the performance evaluation of our protocol, respectively. Section 7 presents the related work. Finally, Section 8 concludes the paper.

2 System Model and Requirements

System Model. As shown in Figure 1, a system model of a collaborative authentication system consists of the following entities:

- *User*: a person who wants to access various services provided by a service provider. The user also carries a number of personal devices (including wearable) which can be used in the authentication procedure.
- *Personal devices*: devices owned by the user. Some of these devices such as a smartphone have secure storage where the user's secret data can

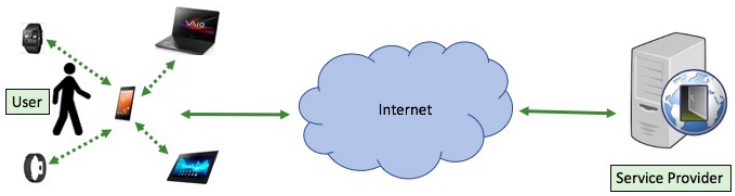

Fig. 1: A system model of a collaborative authentication scheme.

be stored. However, other devices such as wearables (e.g., wristband) may not have such secure storage. Each of the user's devices may also have one or more sensors integrated to measure different (physiological) data such as location, gait, blood pressure, heart beats, etc.
– *Service provider*: a service provider to which users want to authenticate in order to access various services and/or resources.

Functional Requirements. A collaborative authentication system should support the following functional requirements:

– *Collaborative*: the user should use data (input) provided by multiple user devices.
– *Flexible*: the user should be able to use various combinations of data collected from various user devices.
– *Robust and resilient*: a failure or lack of a single user device should not result in an authentication failure.

Security and Privacy Requirements. A collaborative authentication protocol should satisfy the following security and privacy requirements.

– *Multi-factor authentication*: A user should need to use multiple factors (devices) in order to authenticate successfully to the service provider.
– *No authentication key stored*: None of the user's personal devices should store the secret authentication key in its entire form. They should either store shares of the key or generate these shares on demand.
– *Lost and/or stolen device resistance*: Loss of a certain number of a user's devices (less than a specified threshold) should still allow the user to successfully authenticate towards the service provider, but it should not allow an attacker (in possession of those lost devices) to successfully impersonate the user.

- *No biometrics/behaviometrics information stored*: None of the user's personal devices should store any sensitive biometrics and/or behaviometrics information.
- *Device presence protection*: The service provider should not be able to identify which (combination of) personal devices the user uses in the authentication procedure.

3 Building Blocks and their Security Definitions

Secret Sharing. Secret sharing enables the sharing of a secret S among n parties, so that any subset of t or more parties can efficiently reconstruct the secret, and any subset of $t-1$ or less parties learn no information whatsoever about the secret. It is a key technique widely used in secure multiparty computation, which allows n players to compute an agreed function of their inputs in a secure way. Some secret sharing schemes, such as additive secret sharing, can offer security against dishonest majorities, whereas others, such as Shamir's secret sharing, require at least an honest majority [9]. Security in this setting means guaranteeing the correctness of the output as well as the privacy of the players' inputs, even when some players cheat. Shamir's scheme is what is typically called a (t, n) (or t-out-of-n)-threshold scheme where n is the total number of parties involved in the computation and t is the size of the subset needed to reconstruct the secret.

Let P be the set of n parties involved in a computation, and let $\mathbb{Z}_q$ be a finite field and $S \in \mathbb{Z}_q$ a secret. A dealer calculates the shares S_i, $i = 1, \cdots, n$ of secret S using a polynomial $f(x)$ of degree $t-1$, by evaluating $f(i) \mod q$, $\forall i = 1, \cdots, n$, i.e., $S_i = f(i)$.

Definition 1 (Shamir Secret Sharing). *Let $S \in \mathbb{Z}_q$ be a secret to be secretly shared among n parties. Then Shamir's (t, n)-threshold secret sharing scheme works as follows. First, a degree $t-1$ polynomial $f(x) = S + a_1 x + \cdots + a_{t-1} x^{t-1}$, where $a_j \in \mathbb{Z}_q$, $j = 1, \cdots, t-1$ are randomly selected coefficients, is chosen. Then the shares S_i, $i = 1, \cdots, n$, are generated by evaluating $f(x)$ at each i, i.e., $S_i = f(i) \mod q$. Given t shares, $f(x)$ can be reconstructed using interpolation, and then the secret is equal to the evaluation of the polynomial at 0, i.e., $S = f(0) \mod q$.*

Fuzzy Extractor. A fuzzy extractor is a randomness extractor that comprises a pair of procedures known as *generation* and *reproduction* algorithms. The former algorithm generates a random string and helper

data from an input. The latter algorithm reproduces the same random string using the helper data and an input that is *close* to the original input to the generation algorithm. Fuzzy extractors (and also similar concepts such as fuzzy commitments) [10–14] are commonly used as biometric cryptosystems for template protection. Formally, a fuzzy extractor is a construct that must satisfy the following properties [13].

Definition 2 (Fuzzy Extractor). *An (m, ℓ, t, ϵ)-`fuzzy extractor` over a metric space $(\mathcal{M}, d)$ comprises a pair of efficient randomized procedures (`Gen`, `Rep`). The generation algorithm `Gen`, on input $\omega \in \mathcal{M}$, outputs an extracted string $S \in \{0,1\}^{\ell}$ and a helper string $P \in \{0,1\}^{*}$. The reproduction procedure `Rep` takes an element $\omega' \in \mathcal{M}$ and a helper string $P \in \{0,1\}^{*}$ as inputs and outputs S', such that the following two properties are satisfied:*

- *(**Security**) For any distribution W over $\mathcal{M}$ with min-entropy m, S is indistinguishable from a uniformly random string even when conditioned on P. That is, if the min-entropy $\mathbf{H}_{\infty}(W) \geqslant m$ and `Gen`$(W) \rightarrow (S, P)$, then we have*

$$\mathbf{SD}((S, P), (U_{\ell}, P)) \leqslant \epsilon,$$

  *where **SD** is the statistical distance and U_{ℓ} is a uniformly distributed random string of length ℓ.*
- *(**Error-tolerance**) If $d(\omega, \omega') \leqslant t$ and `Gen`$(\omega) \rightarrow (S, P)$, then `Rep`$(\omega', P) = S' = S$. This means that the exact value of S can be reproduced using the helper data P and any new sample ω' which is close to the originally sampled template ω from which S and P were generated.*

Threshold Signatures. The concept of threshold signatures is similar to that of secret sharing. A t out of n threshold signature scheme is one that allows any subset of t players to generate a signature, while disallowing the generation of a valid signature the number of participating players is less than t. A threshold signature scheme $\mathsf{TS} = (\mathsf{TKeyGen}, \mathsf{TSIGN}, \mathsf{VER})$ comprises three algorithms.

- The threshold key generation algorithm $\mathsf{TKeyGen}$ takes t, n and a security parameter λ as input and employs a (t, n) Shamir secret sharing to generate n shares $\mathsf{sk}_1, \cdots, \mathsf{sk}_n$ of a private key sk, and the corresponding public key pk.
- The possibly randomised threshold signature generation algorithm TSIGN further consists of signature share generation and signature

construction algorithms. Each participating player i outputs its signature share upon taking a message m and its share sk_i as input. The signature construction algorithms combines all signature shares and outputs a message-signature pair (m, σ).
– The verification algorithm VER uses the public key pk to verify if a message-signature pair (m, σ) is valid.

Definition 3. *Let a* $\mathsf{TS} = (\mathsf{TKeyGen}, \mathsf{TSIGN}, \mathsf{VER})$ *be a threshold digital signature scheme. Then* TS *is called secure if no probabilistic polynomial-time adversary* $\mathbf{A}$ *that is allowed to corrupt up to* $t-1$ *players can forge a valid signature on any message.*

Threat Model and Assumptions. The threat model used in this paper is the following. *Users* are malicious. They might try passively and/or actively to collect and alter the information stored and exchanged within the authentication system, in an attempt to gain access to services which he or she does not have permission to access. The *service provider* is honest-but-curious. It might try to learn and extract unauthorised information about the users. *External entities* (i.e., third parties) are malicious. They are active adversaries who try to impersonate legitimate users.

Assumption 1 *We assume that users' (i) personal devices (excluding wearables) are equipped with secure storage, security mechanisms to provide access control and protection against data breaches and/or malware, (ii) wearables do not have any secure storage element, and (iii) the devices can communicate with each other.*

Assumption 2 *We assume that the employed threshold signature scheme is secure, cf Definition 3.*

4 The Collaborative Authentication Protocol

In this section, we propose a collaborative authentication protocol that combines a threshold signature scheme and fuzzy extractors. Afterwards, we detail how the remaining shares could be used to regenerate a new share to replace a lost one (due to a lost device). Finally, we propose a new threshold signature scheme by combining the classical Schnorr signature scheme with the Shamir secret sharing scheme.

4.1 Protocol Overview

Below we explain how any threshold signature scheme could be used to support collaborative authentication schemes. Then we describe how these schemes could be combined with fuzzy extractors to allow some of the users' personal devices to generate the shares of the secret key on demand, rather than storing them.

Using Threshold Signatures. Suppose that sk and pk are the private and public key pair of a threshold signature scheme. Then sk is split into shares which are distributed to the personal devices of a user. To sign a challenge during the authentication process, these devices perform computations on the challenge using their shares. Then, the results of these computations are combined (by one of the user's personal devices) to form a valid signature, which will be verified by the service provider to verify the user identity. In particular:

- The user's devices share the private key using secret sharing scheme among themselves. The shares can be generated by one of the personal devices, say, a smartphone, tablet, personal computer of the user.
- To authenticate a user, the service provider sends a challenge to the user's device which initiated the authentication request or acts as a gateway device.
- Upon receiving the challenge, the devices jointly compute a signature without reconstructing the user's private key.
- The signature is sent to the service provider, which verifies the signature using the user's public key.

Using Biometrics/Behaviometrics. The authentication system can also incorporate biometrics or behaviometric information for increased security and flexibility. In this case, the shares of the key can either be accessed by the devices using the measured biometric/behaviometric information, or they can be directly generated from such sources on demand. Below we describe how such fuzzy sources can be utilised to generate the shares.

- A key is generated from the user biometrics using a fuzzy extractor in one of the user's biometric-enabled devices. The generated key will be used as one of the shares of the private key for the signature scheme. The generated public data, named helper data, is stored either in the device itself if there is sufficient storage, or in a gateway device,

which can be one of user's more computationally-capable personal devices, e.g., a smartphone, in the case the device has limited storage capabilities, e.g., a wearable.
– During the authentication procedure, the behaviometric data is used to reproduce the share with the help of the helper data, and then the signature share is computed by the device. In the case of a wearable, the gateway device sends the helper data to the device which then uses it to reproduce the share, which in turn is used for calculating the signature share of the device by the device.

4.2 Share Regeneration and Repair

It may happen from time to time that a user device gets lost or damaged, which results in the loss of the share distributed to that device; or even a new device is added. In such cases, regeneration of the shares of the key using the remaining shares without reconstructing the entire private key becomes necessary. Fortunately, this is possible due to the nature of the Shamir secret sharing. Below we describe the main idea with an example; for details we refer the curious reader to [15].

Suppose that a secret S is shared among n players using the Shamir secret sharing and that the shares are $S_1, \cdots, S_n$. Also, suppose that one share, say, S_i, is lost. We want to generate new shares, $S'_1, \cdots, S'_n$, of the secret S without reconstructing S. The procedure for achieving this is as follows.

– All players, except for the i-th player, generate n shares, say $S_{j1}, \cdots, S_{jn}$, of their share S_j, $j = 1, \cdots, n$ and $j \neq i$.
– The shares of S_j are distributed to all players, so each player has $n-1$ shares of the remaining shares of the original secret S.
– Each player uses t shares to interpolate a polynomial $f_i(x)$ of degree $t-1$ and evaluates the polynomial at 0.
– The new shares of the original secret S is $S'_i = f_i(0)$, for $i = 1, \cdots, n$.

Note that a distributed key generation scheme also follows the same procedures as the share regeneration. The only difference is that in distributed key generation, each and every participating player picks a random secret string and shares it with the rest of the players using a (t, n) secret sharing scheme. Then, using the shares at its disposal, each player locally computes a value that will be a share of a common secret. Later on, we will present a performance analysis for both distributed key generation and share regeneration.

This share regeneration procedure requires communication among all players and the number of messages exchanged among them is $(n-1)^2$. The required computational complexity is also $n \times t$ modular additions and $n(n-1)$ polynomial evaluations.

However, in cases where we only want to recover a single share, then there is a more efficient way to achieve this rather that regenerating the shares of all parties. In this case, we can use what is called an *enrollment repairable threshold scheme (eRTS)* proposed by Stinson and Wei [8]. This eRTS only requires a subset of k players to help repair a share, where $t \leqslant k \leqslant n$; see [16] for more on reparable threshold schemes. Below we describe how the enrollment repairable threshold scheme works.

Suppose that a secret S is shared among n players using a (t, n)-Shamir secret sharing scheme and we wish to repair the share S_ℓ for a player P_ℓ. Assume that t players $P_1, \cdots, P_t$ are helping with the recovery of S_ℓ and that $\ell > k$. Suppose that $S_\ell = f(\ell)$, where $f(x) \in \mathbb{Z}_q[x]$ is a random polynomial of degree at most $t-1$ whose constant term is the secret S. The share S_ℓ can be expressed as

$$S_\ell = \sum_{i=1}^{t} \omega_i s_i,$$

where the ω_i's are public Lagrange coefficients. The eRTS proceeds as follows:

- $\forall\, 1 \leqslant i \leqslant t$, player P_i picks random values δ_{ij}, $j = 1, \cdots, t$ such that

$$\sum_{j=1}^{t} \delta_{ij} = \omega_i S_i.$$

- $\forall\, 1 \leqslant i \leqslant t$, $1 \leqslant j \leqslant t$, player P_i sends δ_{ij} to player P_j.
- $\forall\, 1 \leqslant j \leqslant t$, player P_j computes

$$\sigma_j = \sum_{i=1}^{t} \delta_{ij}.$$

- $\forall\, 1 \leqslant j \leqslant t$, player P_j sends σ_j to player P_ℓ.
- Player P_ℓ recovers S_ℓ by computing

$$S_\ell = \sum_{j=1}^{t} \sigma_j.$$

As can be seen, the eRTS requires only an exchange of t^2 messages and $2t^2 - t - 1$ modular additions. The communication complexity can further be reduced to $t(t+1)/2$ messages and the computational complexity to $t(t+1)/2$ modular additions by requiring that player P_i does not send anything to player P_j if $i < j$, as shown in [16].

4.3 Threshold Schnorr Signatures

As a concrete example, here we present a threshold signature scheme that can be employed in the collaborative authentication protocol proposed in this paper. Namely, we present threshold Schnorr signatures which can be obtained from the classical Schnorr signatures by using secret sharing.

Schnorr Signature Scheme. Let q be a prime. Let $\mathbb{G}$ be a group of order q in which the discrete log is (assumed to be) hard, and let g be the generator of $\mathbb{G}$. Let M be the message space. Also, let $H : \{0,1\}^* \mapsto \{1, 2, \cdots, q-1\}$ be a cryptographic hash function. The Schnorr signature scheme, which is constructed by applying the Fiat-Shamir transformation [17] to Schnorr's identification protocol [18], works as follows.

The private key sk is an integer x in the range $1 \leqslant x \leqslant q-1$, and the public key pk is $y = g^x \mod q$.

- **Sign:** $(r, s) \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$: A message $m \in M$ is signed using the secret key sk by first picking a random integer k in the range $1 \leqslant k \leqslant q-1$; then $r = g^k \mod q$ and $s = H(m||r)x + k \mod q$. Here $m||r$ is concatenation of bit representation of m and r.
- **Verify:** $1/0 \leftarrow \mathsf{Verify}(\mathsf{pk}, m, s, r)$: To verify whether the signature (s, r) for m is correct, use the public key y to check if $g^s \stackrel{?}{=} y^{H(m||r)}r \mod q$.

Threshold Schnorr Signature Scheme. The threshold scheme has a share combining algorithm which takes as input a message and t valid signature shares on the message, along with the public key, and outputs a valid signature on the message.

Suppose that there are $n \geqslant 2$ players, and that the parameters, that is, the group $\mathbb{G}$, the private secret x and the public key $(g, q, y = g^x \mod q)$, are the same as before. The secret x is partitioned into n shares $x_1, x_2, \cdots, x_n$ using an additive secret sharing scheme and distributed to the respective parties. The shares are such that at least t distinct shares would allow the reconstruction of the secret x.

Now, in order to sign a message m, each party randomly picks a random integer k_i with $1 \leqslant k_i \leqslant q-1$ and provides $g^{k_i} \mod q$ to one party, which we call the central party, which then computes $r = \prod_{i=1}^{t} g^{k_i} \mod q$ and $h = H(m||r)$, which then is sent back to each party. The devices provide $s_i = hx_i + k_i \mod q$ to the central party, which combines them to generate the signature (s, r) on the message m. So in summary, the threshold Schnorr signatures work as follows.

- The private key x is shared among the devices using Shamir secret sharing.
- Suppose that t devices are present, and let G be the set of those devices.
- To each device $i \in G$, the gateway device sends $\omega_i = \prod_{\substack{j \in G \\ j \neq i}} \frac{j}{i-j}$.
- Each device $i \in G$, picks a random k_i and provides $r_i = g^{\omega_i k_i}$ to the gateway.
- The gateway computes $r = \prod_{i \in G} r_i = g^{\sum_{i \in G} \omega_i k_i}$ and $h = H(c||r)$ and sends h to all devices in G.
- Each device $i \in G$ provides the gateway with $s_i = k_i + hx_i$
- Finally, the signature is (r, s) where

$$s = \sum_{i \in G} s_i \omega_i = \sum_{i \in G} (k_i + hx_i)\omega_i = \sum_{i \in G} k_i \omega_i + h \sum_{i \in G} x_i \omega_i = \sum_{i \in G} k_i \omega_i + hx.$$

Note that in threshold signature schemes the signing key is not reconstructed to sign a message. Instead the players compute signature shares using their share of the signing key, and the signature shares are combined to form a signature on the message.

5 Security Analysis

Our collaborative authentication protocol satisfies all the security and privacy requirements specified in Section 2. More specifically and informally speaking, it provides a multi-factor authentication as it requires the presence (and collaboration) of multiple user devices due to the use of secret sharing scheme. A user will not be able to authenticate successfully towards the service provider if he/she uses only one of his/her personal devices. For the same reasons, the protocol also does not require the user's private signature key to be stored in any of the user's personal devices. Instead, these devices only store (or generate) shares of the key.

In addition, the use of a threshold scheme allows the protocol to provide lost/stolen user device protection as well as device presence protection.

More specifically, the nature of threshold schemes - the fact that not all of the shares, but only a subset of these shares, are required to perform a specific operation successfully - allows users to successfully generate a valid signature even if one or more of his/her devices (depending on the specified threshold) are not present due to loss or theft. For the same reason and for the fact that the user's signature is constructed locally (in one of the personal devices) and then sent to the service provider, our protocol provides protection against leakage of information about device presence. As a valid signature could be constructed using various combinations of subset of the user's personal devices, the service provider does not know exactly which devices the user carries at the time of the authentication procedure. Moreover, the use of fuzzy extractors allows users to generate some of the shares of the key without the need of storing any biometric or behaviometric information.

Formally speaking, it is obvious that for the proposed collaborative authentication protocol to be secure, the underlying threshold signature scheme has to be secure (i.e., unforgeable). There are threshold signature schemes that satisfy various security requirements, such as the practical threshold RSA signatures by Shoup [19] and more recent ones by Simoens et al. [15]. Shamir secret sharing is used in the former and verifiable secret sharing together with bilinear maps in the latter. In addition, Schnorr signatures are unforgeable and the unforgeability of (t, n)-threshold Schnorr signatures is also straightforward assuming that no more than $t - 1$ players are corrupted.

Below, we first give a security definition for a collaborative authentication protocol, and then show that our protocol is secure.

Definition 4. *We say that a collaborative authentication protocol using a (t, n)-threshold signature scheme is secure against active attacks if for all efficient (i.e., probabilistic polynomial time) adversaries* $\mathbf{A}$ *that can corrupt up to $t - 1$ players, the advantage of* $\mathbf{A}$ *in the following game between a Challenger and* $\mathbf{A}$ *is negligible.*

- Key generation. *The Challenger runs* $(\mathsf{pk}, \mathsf{sk}_1, \cdots, \mathsf{sk}_n) \leftarrow \mathsf{TKeyGen}(\lambda, n, t)$, *and sends* pk *and a randomly chosen $t - 1$ shares of* sk *to* $\mathbf{A}$.
- Active attack phase. *The adversary interacts with the prover (i.e., the user) and gets the prover to produce signatures for a polynomial number of challenges. In this case,* $\mathbf{A}$ *plays the role of the verifier and the Challenger that of the prover.*
- Impersonation attempt. *The Challenger and* $\mathbf{A}$ *interact with* $\mathbf{A}$ *playing impersonating the prover. The Challenger requests* $\mathbf{A}$ *to produce a signature for a challenge c, and* $\mathbf{A}$ *responds with (c, σ).*

The adversary wins the game if σ is a valid signature for c. The adversary's advantage is defined as the probability that **A** *wins the game.*

Theorem 1 (Security). *Let M be the space of all authentication challenges, and assume that the size $|M|$ of M is super-poly. Assume further that Assumption 1 and Assumption 2 holds. Then the presented collaborative authentication protocol using threshold signatures is secure against active attacks as defined in Definition 4.*

Proof (Sketch). First of all, the assumption that the size of the challenge space $|M|$ is super-poly implies that in each impersonation attempt, the probability that **A** gets challenge that it has previously asked the prover in the attack phase is negligible. Assumption 1 is necessary for security for obvious reasons. To prove that Assumption 2 (i.e., the assumption that the threshold signature scheme is unforgeable) implies the security of the authentication protocol, we show that a successful attack on the authentication protocol can be converted in a blackbox way into a successful attack on the signature scheme. This is also straightforward. Suppose that an adversary **A** can break the authentication protocol. This means that **A** can generate a valid signature on an authentication challenge. Now, suppose that another adversary $\mathbf{A}'$ is attempting to forge a valid signature for a message. Then $\mathbf{A}'$ first simulates the authentication protocol of **A** using the threshold signature scheme. Then $\mathbf{A}'$ presents **A** with the message as an authentication challenge. **A**'s response will be a valid signature (for the message) that $\mathbf{A}'$ was attempting to forge.

Therefore, either the adversary gets the same challenge that it previously used in the attack phase, or it successfully forges a signature. The probability for both is negligible. □

In the case of using fuzzy extractors for generation of some of the shares, the fuzzy extractor must satisfy Definition 2. The fuzzy extractors introduced by Dodis et al. [14] already satisfy that definition. Security of fuzzy extractors also implies the privacy of behaviometric data, as the public helper data does not reveal information on the behaviometric data or the extracted key. Lastly, the share regeneration procedure is information-theoretically secure against honest majority, as it is the Shamir secret sharing, cf. Definition 1.

6 Performance Analysis

To evaluate the performance of our authentication protocol, we first have prototyped and tested our basic scheme for share regeneration and

Table 1: CPU running times.

Protocol	Time in Seconds	Key Sizes
Key Generation	3.57×10^{-4}	1024
Share Reconstruction	7.60×10^{-5}	1024

distributed key generation on relatively small key sizes (1024 bits). We implemented our test cases using C++ and the secret sharing tools (Shamir Secret Sharing [9]) implemented by the *mpcToolkit*[1] introduced in [20]. The library was implemented over NTL (Library for doing Number Theory) [21] and natively supports 63-bit inputs. Hence, we further adapted it to support longer key sizes. We run our tests over a $2 \times 2 \times 10$-cores Intel Xeon E5-2687 server at 3.1 GHz. We simulated a three-device scenario over our server, and averaged the results of 10000 executions. The computational times are depicted in Table 1. As can be seen from the table, both procedures for distributed key generation and share regeneration are efficient.

We then tested the performance of a python implementation of threshold Schnorr signatures in a simulated network environment on the same server. With parameters for 128-bit level security (chosen according to ECRYPT II recommendations [22]), the total runtime for calculating the signature shares, communicating them and combining them to form a signature is approximately 0.022 seconds, for 3-out-of-5 devices.

7 Related Work

Shamir [9] was the first to introduce the concept of secret sharing. Feldman [23] extended this concept by introducing verifiable secret sharing. Pedersen [24] then used this idea to construct the first Distributed Key Generation (DKG) protocol. To increase the resilience in threshold schemes, the number of devices included in the scheme should be maximized. Therefore, Simoens et al. [15] presented a new DKG protocol and demonstrated how this allows wearables not capable of securely storing secret shares to be incorporated. Peeters et al. [25] used this idea to propose a threshold distance bounding protocol. In this paper, we consider a threshold-based authentication protocol, where the secret key is shared among a set of user devices such as mobile phone and wearables.

Ever since the introduction of the general notion of threshold signatures by Desmedt [26], threshold signature schemes have been extensively studied

[1] `https://github.com/abdelrahamanaly/mpcToolkit`.

in the literature. For example, Desmedt and Frankel [27] and Harn [28] respectively presented a non-robust and a robust threshold ElGamal signature scheme [29] based on secret sharing [9]. These schemes have small share size and require synchronized interaction among the players. Gennaro et al. [30] presented a robust threshold digital signature standard scheme. Shoup [19] showed how RSA signatures could be transformed into a robust and practical threshold-based variant.

On the collaborative continuous authentication front, there have been a recent surge in interest and the industry demand due to the significantly increased use of smartphones, tablets and wearables. Martinovic et al. [31] introduced a new biometric based on the human body's response to an electric square pulse signal, called pulse-response. It is proposed to enhance security in the context of two example applications: (1) an additional authentication mechanism in PIN entry systems, and (2) a means of continuous authentication on a secure terminal. As the authors show, the pulse-response biometric is effective because each human body exhibits a unique response to a signal pulse applied at the palm of one hand, and measured at the palm of the other. Patel et al. [32] gave a nice survey on continuous authentication on mobile devices, and also discussed challenges and research directions in this field. This survey along with the references therein provides sufficient information on the state-of-the-practice in this field of continuous authentication. Van hamme et al. [2] reviewed emerging trends and challenges with collaborative frictionless authentication systems and identified the enrollment of users, usability as well as security and privacy of such systems as key research challenges. Mustafa et al. [33] provided a comprehensive threat analysis of such a system and specified a list of security and privacy requirements. The authors also suggested three high-level solutions to address these requirements, however, without providing any specific design or implementation details.

8 Conclusions

In this paper, we proposed a collaborative authentication protocol based on the use of threshold signature schemes. In particular, the user devices store the shares of the signing key (i.e., the private key) of a threshold signature scheme, and the private key is never reconstructed to minimise the security threats. In case a share of the key is lost, the remaining shares can be used to (i) generate new shares of the key for each party or (ii) repair only the lost share without reconstructing the entire key. Furthermore, we showed how the shares of the key can also be generated

using contextual and/or behavioral information using fuzzy extractors. Finally, our performance results demonstrate the practical feasibility of our collaborative authentication protocol.

Acknowledgments. We thank the anonymous reviewers for their valuable comment. This work was supported by funded by imec through ICON DiskMan, the Security & Privacy Centre projects on Biometrics & Authentication and Secure Distance Bounding. It was also funded by the Flemish government through the FWO SBO project SPITE S002417N.

References

1. Sagiroglu, S., Sinanc, D.: Big data: A review. In: International Conference on Collaboration Technologies and Systems (CTS'13). (2013) 42–47
2. Van hamme, T., Rimmer, V., Preuveneers, D., Joosen, W., Mustafa, M.A., Abidin, A., Argones Rúa, E.: Frictionless authentication systems: Emerging trends, research challenges and opportunities. In: the 11th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'17), IARIA (2017)
3. Bhargav-Spantzel, A., Squicciarini, A., Bertino, E.: Privacy preserving multi-factor authentication with biometrics. In: Proceedings of the Second ACM Workshop on Digital Identity Management. DIM '06, New York, NY, USA, ACM (2006) 63–72
4. Bonneau, J., Herley, C., Oorschot, P.C.v., Stajano, F.: The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy. SP '12, Washington, DC, USA, IEEE Computer Society (2012) 553–567
5. Grosse, E., Upadhyay, M.: Authentication at scale. IEEE Security Privacy **11**(1) (Jan 2013) 15–22
6. Guidorizzi, R.P.: Security: Active authentication. IT Professional **15**(4) (July 2013) 4–7
7. Preuveneers, D., Joosen, W.: Smartauth: Dynamic context fingerprinting for continuous user authentication. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing. SAC '15, New York, NY, USA, ACM (2015) 2185–2191
8. Stinson, D.R., Wei, R.: Combinatorial repairability for threshold schemes. Designs, Codes and Cryptography **86**(1) (2018) 195–210
9. Shamir, A.: How to share a secret. Communications of the ACM **22**(11) (1979) 612–613
10. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: ACM Conference on Computer and Communications Security, ACM (1999) 28–36
11. Juels, A., Sudan, M.: A fuzzy vault scheme. IACR Cryptology ePrint Archive **2002** (2002)
12. Juels, A., Sudan, M.: A fuzzy vault scheme. Des. Codes Cryptography **38**(2) (2006) 237–257
13. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: EUROCRYPT. Volume 3027 of LNCS., Springer (2004) 523–540
14. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1) (2008) 97–139

15. Simoens, K., Peeters, R., Preneel, B.: Increased resilience in threshold cryptography: Sharing a secret with devices that cannot store shares. In: International Conference on Pairing-Based Cryptography. Volume 6487 of LNCS., Springer (2010) 116–135
16. Laing, T.M., Stinson, D.R.: A survey and refinement of repairable threshold schemes. eprint:2017/1155
17. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Conference on the Theory and Application of Cryptographic Techniques, Springer (1986) 186–194
18. Schnorr, C.: Efficient identification and signatures for smart cards. In: CRYPTO'89, Springer (1990) 239–252
19. Shoup, V.: Practical threshold signatures. In: EUROCRYPT'00. Volume 1807 of LNCS., Springer (2000) 207–220
20. Aly, A.: Network Flow Problems with Secure Multiparty Computation. PhD thesis, Universté catholique de Louvain, IMMAQ (2015)
21. Shoup, V.: NTL: A library for doing number theory (2001)
22. ECRYPT II NoE: ECRYPT II yearly report on algorithms and key lengths (2011-2012) (2012) ECRYPT II deliverable D.SPA.20-1.0.
23. Feldman., P.: A practical scheme for non-interactive verifiable secret sharing. In: FOCS'87, IEEE Computer Society (1987) 427437
24. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: CRYPTO'91. Volume 576 of LNCS., Springer (1992) 129140
25. Peeters, R., Singelee, D., Preneel, B.: Toward more secure and reliable access control. IEEE Pervasive Computing **11**(3) (2012) 76–83
26. Desmedt, Y.: Society and group oriented cryptography: a new concept. In: CRYPTO'87. (1987) 120–127
27. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: CRYPTO'89. (1989) 307–315
28. Harn, L.: Group-oriented (t, n) threshold digital signature scheme and digital multisignature. IEE Proceedings-Computers and Digital Techniques **141**(5) (1994) 307–313
29. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory **31**(4) (1985) 469–472
30. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold dss signatures. Information and Computation **164**(1) (2001) 54–84
31. Rasmussen, K.B., Roeschlin, M., Martinovic, I., Tsudik, G.: Authentication using pulse-response biometrics. In: NDSS. (2014)
32. Patel, V.M., Chellappa, R., Chandra, D., Barbello, B.: Continuous user authentication on mobile devices: Recent progress and remaining challenges. IEEE Signal Processing Magazine **33**(4) (2016) 49–61
33. Mustafa, M.A., Abidin, A., Argones Rúa, E.: Frictionless authentication system: Security & privacy analysis and potential solutions. In: the 11-th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'17), IARIA (2017)