DAMFA: Decentralized Anonymous Multi-Factor Authentication

Omid Mir
Johannes Kepler University Linz
LIT Secure and Correct Systems Lab
Linz, Austria
mir@ins.jku.at

Michael Roland
Johannes Kepler University Linz
Institute of Networks and Security
Linz, Austria
roland@ins.jku.at

René Mayrhofer
Johannes Kepler University Linz
Institute of Networks and Security
Linz, Austria
rm@ins.jku.at

ABSTRACT

Token-based authentication is usually applied to enable single-sign-on on the web. In current authentication schemes, users are required to interact with identity providers securely to set up authentication data during a registration phase and receive a token (credential) for future accesses to various services and applications. This type of interaction can make authentication schemes challenging in terms of security and usability. From a security point of view, one of the main threats is the compromisation of identity providers. An adversary who compromises the authentication data (password or biometric) stored with the identity provider can mount an offline dictionary attack. Furthermore, the identity provider might be able to track user activity and control sensitive user data. In terms of usability, users always need a trusted server to be online and available while authenticating to a service provider.

In this paper, we propose a new Decentralized Anonymous Multi-Factor Authentication (DAMFA) scheme where the process of user authentication no longer depends on a trusted third party (the identity provider). Also, service and identity providers do not gain access to sensitive user data and cannot track individual user activity. Our protocol allows service providers to authenticate users at any time without interacting with the identity provider.

Our approach builds on a Threshold Oblivious Pseudorandom Function (TOPRF) to improve resistance to offline attacks and uses a distributed transaction ledger to improve usability. We demonstrate practicability of our proposed scheme through a prototype.

CCS CONCEPTS

• **Security and privacy → Cryptography**; **Distributed systems security**; **Privacy-preserving protocols**.

KEYWORDS

Multi-factor authentication, Blockchain, Oblivious pseudorandom function, Usability, User anonymity

BSCI '20, October 6, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7610-5/20/10…$15.00
https://doi.org/10.1145/3384943.3409417

ACM Reference Format:
Omid Mir, Michael Roland, and René Mayrhofer. 2020. DAMFA: Decentralized Anonymous Multi-Factor Authentication. In *Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure (BSCI '20), October 6, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3384943.3409417

1 INTRODUCTION

Single-Factor Authentication Key Exchange schemes. Authenticated Key Exchange (AKE) is one of the most broadly used cryptographic primitives which enables two parties to create a shared key over a public network. Typically, the parties need to have authentication tokens, e.g., cryptographic keys (public or secret) or short secret values (i.e., low-entropy). They also securely store these authentication tokens in a trusted service provider during the registration phase. There are various types of authentication factors such as knowledge, possession, and physical presence; low-entropy passwords are widely present in practice. An authentication protocol that uses a password is a Password based Authentication Key Exchange (PAKE). Each service provider creates an account for the user and holds the user's account information (e.g., username, password hash). However, passwords are usually vulnerable to both online and offline attacks [11, 49]. An attacker who compromises the data stored (i.e., salted password hashes) in the server can run an offline dictionary attack. This attack leads to the disclosure of user accounts [11, 18, 21]. Even if low entropy passwords are correctly salted and hashed, they still do not resist against the brute force of modern hardware. In 2012, a rig of 25 GPUs could test up to 350 billion guesses per second in an offline dictionary attack [37].

Multi-Factor Authentication Key Exchange schemes. PAKE schemes have become less secure than expected due to the low entropy of passwords and increasing attacker ability. Multi-Factor Authentication (MFA) schemes overcome this risk by adding additional authentication factors. MFA is widely recognized as a method which combines a password with a secret value and stores it in tokens (e.g., two-factor SSH with USB sticks). In addition, recent advances by fingerprint readers and other sensors lead to increased usage of smartphones and biometric factors in MFA schemes (see Fig. 1). However, existing MFA schemes incorporate password authentication and 2nd-factor authentication as separate mechanisms leading to different vulnerabilities such as spoofing and offline attacks [29, 35]. Indeed, this kind of authentication scheme stores a (salted) one-way hash function of the password on the server. Therefore, an adversary who compromises the server and learns the password hash can still recover the password [29]. Moreover, users need to enroll with service providers by using mobile devices (USB tokens and more) with the associated security and privacy concerns. These

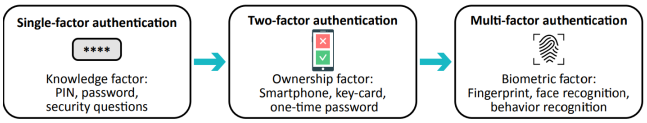

Figure 1: Evolution of authentication methods from SFA to MFA [35]

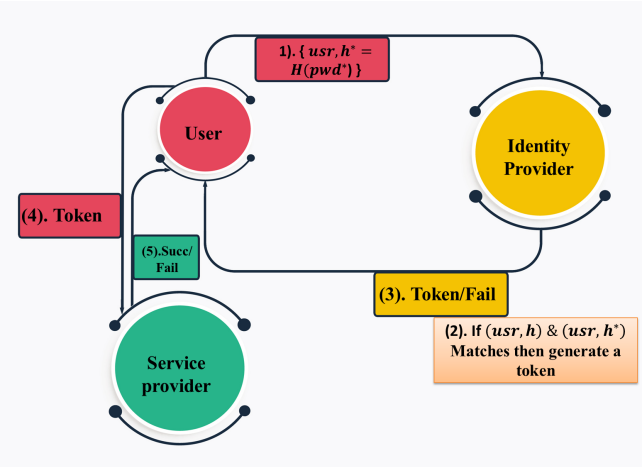

Figure 2: The generic flow diagram shows the authentication phase of a password-based token method. The figure does not include the registration phase where the user stores its username (usr) and the hash of its password in the identity provider. This is taken from Agrawal et al. [1].

devices are more prone to theft and loss due to their mobility. Also, malware can infect them.

Anonymous authentication. In general, authentication schemes do not protect user privacy. User's activities in the digital world can easily be logged and analyzed. Leakage of individual information may have serious consequences for users, e.g., financial/credit losses. To meet the increasing need of privacy protection, we need to reinforce multi factor authentication with the preservation of user privacy. An anonymous authentication scheme is a protocol that allows a member called a prover of a legitimate group to convince a verifier that she is a member of the group without revealing any information about herself. Recently, a few schemes for anonymous password authentication [32, 45, 48, 50] have been proposed. In particular, anonymous password authentication promises unlinkability, i.e., the server should not be able to link user accesses, such that logins from the same user cannot be recognized as such.

Building a fully decentralized authentication architecture. An identity provider (IDP) with centralized databases of authentication data of all individuals could easily facilitate the MFA scheme and present more convenience for individuals by providing a Single-Sign-On (SSO) method [41]. For example, several initiatives (centralized systems) like Token (JWT) [3], OAuth [24], SAML [36] and OpenID let service providers authenticate users without taking on the responsibility of storing and managing passwords. In all these works, the authentication phase is pretty much the same [1]: In these systems, in the registration phase, a user gives credentials (ID/password) to the user identity provider (a trusted server) which stores the username together with its hashed passwords. In the authentication phase, IDP verifies the client credential by matching the hash during the sign-in process before issuing an authentication credential (a digital signature or a message authentication code) using a master secret key and identifies the user to the websites (or other service providers) she visits (see Fig. 2). However, this kind of centralized system represents several challenges:

- This identity provider represents a single point of failure and an obvious target for compromise. These concerns can severely damage the reliability of the system [1]:
  - Extract the secret key and forge tokens, which enable access to arbitrary services and data in the system.
  - Capture hashed passwords (or biometric) and run an offline dictionary attack to recover user credentials.
- These identity providers can track user activity, leading to serious privacy issues [2, 13].
- There is no authentication solution that allows users to manage and store their digital identity completely by themselves. So, users always need an available identity provider that offers identity management systems (active verification).

To address the above challenges, we build a practical decentralized multi-factor authentication, where the process of user authentication no longer depends on a trusted third party (decentralization property). Subsequently, it does not require IDP to be online while verification (passive verification). Moreover, users do not need to register with each service provider individually (Single-Sign-On property) while preserving user privacy (anonymity).

1.1 Our Contribution

We design a new architecture and protocol for a multi-factor anonymous authentication key exchange system in a decentralized setting, called Decentralized Anonymous Multi-Factor Authentication (DAMFA). This scheme does not rely on a trusted server. Our protocol reduces the unwanted trust into third parties (e.g., a service server or registration centers) by adding a shared ledger and prevents a single point of failure, which can be compromised by malicious attacks. Also, it does not enable user tracking by a third party (since nobody has control over the sensitive user data). Our protocol provides passive verification which means service providers can authenticate users at any time without requiring additional information (interactions) from an identity provider beyond what is available on the shared ledger. This property removes the cost of running secure channels between the service provider and the identity provider as well as the need for online and available entity. Besides, users generate a secret key with service providers through the use of personal identity agents as auxiliary devices that assist the user in the authentication process and prevent offline attacks. In practice, an attacker has to guess the sensitive user data (password and biometric) and also corrupt the user's agents to authenticate (see section 4). Thus, our proposed scheme provides a secure, privacy-friendly authentication protocol for users and service providers. In the implementation section, we demonstrate that our protocol is efficient and practical.

1.2 Organization of this paper

After recalling some related works in the next section. We review the cryptographic primitives used across this paper in section 3. In section 4, we define the notion of a decentralized anonymous authentication scheme and provide a system model, security definition, an overview of our scheme, and construction in sections 4.1,4.2,4.3, and 4.4, respectively. Finally, in section 5, we propose an implementation and performance of a prototype realizing our authentication system.

2 RELATED WORK

Single-Factor (Password) Authentication Key Exchange. For a long time, researchers related to authenticated key exchange protocols were influenced by single-factor authentication in which a password was elected among various factors. Bellovin and Merritt [5] proposed the first scheme, called Encrypted Key Exchange (EKE). In their scheme, a client shares a password with a server and exchanges encrypted information to generate a common session key. EKE was followed by several works [6, 8, 22]. All of these works were proposed without provable security until Bellare, Pointcheval, and Rogaway expanded a general formal model for Password Authentication Key Exchange (PAKE) [4]. After that, two generic schemes of PAKE were proposed in [23] and [20] which are from the most efficient ways of constructing PAKE in the standard model. Recent work explicitly contains a verifier in authenticated key exchange [7]: In this system, verifiers are usually hash values or transformations $V = H(s, \text{pw})$ of the secret passwords pw with public data s called salt. Then, the server stores the pair (s, V) for each user.

Multi-Factor Authentication. The stability of Multi-Factor Authentication (MFA) relies on the reason that if a user has many authentication factors, then it is difficult for an adversary to compromise all of them. The most significant combinations are long-term passwords with secret keys, possibly stored in tokens (e.g., two-Factor SSH with USB sticks). The work of [46] introduced a framework to analyze these two-factor authentication protocols. In the above works, the participants are a user, a client (say a web browser), a server, and a device (say a smartphone). In the authentication phase, the user sends a password and some additional information provided by the device. In most existing solutions, including [9, 34, 46], during the registration process, the user gets a value, called the token, while the server records a hashed password. During the authentication phase, the two required factors (the password and the token) are sent to a verifier. Jarecki et al. [28] provided a device enhanced password-authenticated key exchange protocol employing mobile device storage as a token. This setting serves two purposes: Firstly, for an adversary to be able to mount an offline dictionary attack successfully, he must corrupt the login server in addition to the mobile device storage. Secondly, to login, the user must access the mobile device storage. Another popular factor which has been used to authenticate a client/user to a remote server is biometrics [25, 26, 33, 40]. Fleischhacker et al. [15] also proposed a modular framework (p, q, t)-MFAKE which models biometrics as follow the liveness assumption of [40]. However, Zhang et al. [52] demonstrated that their scheme could not protect privacy. Indeed, biometric authentication becomes a weak point, when the framework directly uses biometric templates. In addition, it requires to respectively execute a lot of sub-protocols which makes the scheme inefficient.

Anonymous authentication. Another authentication approach is an anonymous password authentication protocol that was proposed by Viet et al. [48] by combining an oblivious transfer protocol and a password-authenticated key exchange scheme, which was developed later in [33, 45, 50]. In fact, an anonymous authentication protocol permits users to authenticate themselves without disclosing their identity and becomes an important method for constructing privacy-preserving in public channels. Recently, Zhang et al. [52] presented a new protocol that relies on a fuzzy extractor. They consider a practical application and suggest several authentication factors such as passwords, biometrics (e.g., fingerprint) and hardware with reasonably secure storage (e.g., smartphone).

3 BUILDING BLOCKS

3.1 Pointcheval and Sanders signature

We first recall the credentials scheme proposed by Pointcheval and Sanders [39]. The scheme works in a bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of type 3, with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

- $\text{Setup}(1^\lambda) \rightarrow (\text{params})$: Choose a bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ with order p, where p is a prime number. Let g_1 be a generator of $\mathbb{G}_1$, and g_2 a generator of $\mathbb{G}_2$. The system parameters are $\text{params} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2)$.
- $\text{KeyGen}(\text{params}) \rightarrow (sk, vk)$: Choose a random secret key $sk = (x, y) \in Z_p$. Parse $\text{params} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2)$, and publish the verification key $vk = (g_2, X, Y) = (g_2, g_2^x, g_2^y)$.
- $\text{Sign}(\text{params}, sk, m) \rightarrow (\sigma)$: Parse $sk = (x, y)$. Pick a random element $h \in \mathbb{G}_1$, and output $\sigma = (h, s) = (h, h^{x+y \cdot m})$.
- $\text{Verify}(pk, m, \sigma)$: This algorithm parses σ as (σ_1, σ_2) and checks whether $\sigma_1 \neq 1_{G_1}$ and $e(\sigma_1, X \cdot Y^m) = e(\sigma_2, g_2)$ are both satisfied. In the positive case, it outputs 1, and 0 otherwise.

The signature $\sigma = (h, s)$ is randomizable by choosing a random $r' \in Z_p$ and computing $\sigma' = (h^{r'}, s^{r'})$. The above scheme can be modified to obtain signature on a hidden message (commitment). They also offers a protocol to show a zero-knowledge proof of a signature $\sigma = (\sigma_1, \sigma_2)$.

3.2 Oblivious PseudoRandom Function (OPRF)

A Psuedorandom Function (PRF) F is a function that takes two inputs: a secret function key k and an input x to compute on and outputs $F_k(x)$. A function picked randomly from a PRF family which is secure if it is distinguishable from a random function with the same domain and range with a negligible probability for all PPT distinguishers. An Oblivious PRF (OPRF) [30] is a protocol between two parties (a sender and a receiver) that securely computes $F_k(x)$ where both x and k are the inputs of sender and receiver, respectively, such that no party learns anything except for the input holder that learns $F_k(x)$.

A Threshold OPRF (TOPRF) [27] is an extension of the OPRF which allows a group of servers to secret-share a key k for PRF F with a shared PRF evaluation protocol which lets the user compute $F_k(x)$ on her input x, s.t. both x and k are secret if no more than t of n servers are corrupted (see Fig. 3).

3.3 Secret Sharing scheme

It consists of two PPT algorithms [44]: First, TSSGen generates n shares of the secret key K as $\langle k_1, \ldots, k_n \rangle \leftarrow \mathsf{TSSGen}(K)$, and second TSSRecon uses t shares to retrieve the primary secret value K as $K \leftarrow \mathsf{TSSRecon}(s_1, \ldots, s_t)$. The security is that less than the threshold shares does not theoretically provide any information about the original secret.

3.4 Public Append-Only Ledger

A ledger allows us to keep a list of public information and maintains the integrity of the dataset. It guarantees a consistent view of the ledger for every party. Every user can insert information into the ledger, and when the data is uploaded, nobody can delete or modify them. Moreover, the ledger investigates the correctness of the pseudonym and guarantees that no one can impersonate another participant to release information. Furthermore, it can prepare everyone with the latest data. In this paper, we assume this assumption holds and construct our system on the blockchain technique as a public append-only ledger. There are already some works constructing advanced applications based on this assumption [16, 17, 42]. Yang et al. [51] formally define a public append-only ledger for constructing our DAMFA system (see Fig. 4).

3.5 Zero-knowledge proof of knowledge

In a zero-knowledge proof of knowledge system [43], a prover proves to a verifier that they possess the witness for a statement without revealing any additional information. In this paper, we will use non-interactive zero knowledge proofs known as Fiat-Shamir heuristic [14] as they have the advantage of being non-interactive. For example, NIZKPoK denotes a non-interactive zero-knowledge proof of the elements x and y as $\mathsf{NIZKPoK}\{(x, y) : h = g^x \wedge c = g^y\}$

Key and server initialization. Random key $k \leftarrow Z_p$ is secret shared using Shamir's scheme with parameters n, t; each server $S_i, i \in [n]$, receives share k_i. Also, they consider a hash functions as $H_g : M \rightarrow \mathbb{G}$. Threshold oblivious computation of $F_k(x)$:

- On input x, user U picks $r \leftarrow Z_p$ and computes $A := H_g(x)^r$; it chooses a subset SR of $[n]$ of size $t + 1$ and sends to each server $S_i, i \in \mathsf{SR}$, the value A and the subset SR.
- Upon receiving the message A from U, server S_i verifies that $A \in \mathbb{G}$ and if so it responds with $b_i := A^{\lambda_i \cdot k_i}$ where λ_i is a Lagrange interpolation coefficient for index i and index set SR.
- When U receives messages b_i from each server $S_i, i \in \mathsf{SR}$, U outputs as the result of $F_k(x)$ the value $H(x, (\prod_{i \in \mathsf{SR}} b_i)^{1/r})$.

Figure 3: Protocol TOPRF [27]: (n, t)-threshold computation

F_{B} executes the following steps with parties $\{PA_1, \ldots, PA_n\}$ and an ideal adversary $\mathcal{S}$ as follows:

- Initialize. Initialize creates an empty list L_p in the beginning.
- Store. On input $(\mathsf{Store}, PA_i, \mathsf{Nym}_u^o, M)$, checks that Nym_u^o is a valid pseudonyms for PA_i, then stores the tuple (Nym_u^o, M) to L_p and declares $\mathcal{S}$ that a new item is appended to the list L_p.
- Retrieve. On input$(\mathsf{Retrieve}, PA_i)$, returns the list L_p to PA_i

Figure 4: Functionality F_B [51]

that satisfies both $h = g^x$ and $c = g^y$. Values (x, y) are assumed to be hidden to the verifier. Similarly, the extension can admit a message as input, thus it is also called signature proof of knowledge denoted as $\mathsf{ZKSoK}[m]\{(x, y) : h = g^x \wedge c = g^y\}$.

3.6 Dynamic accumulator

Dynamic accumulator is a primitive allowing a large set of values to be accumulated into a single quantity, the accumulator; and for each value, there exists a witness, which is the evidence attesting that the value is indeed contained in the accumulator. The proof of showing that a value is accumulated in an accumulator can be zero-knowledge, which reveals nothing to the verifier on the value and the witness. A concrete construction of dynamic accumulator is due to [10], with the five algorithms (AccSetup, AccAdd, AccUpdate, AccWitUpdate, and AccVerify) as follows:

- $\mathsf{AccSetup}(1^k, n)$: creates an accumulator key pair (sk_A, pk_A), an empty accumulator acc_ϕ (for accumulating up to n values) and an initial state state_ϕ.
- $\mathsf{AccAdd}(sk_A, i, \mathrm{acc}_V, \mathrm{state}_U)$: allows the authority to add i to the accumulator. It outputs a new accumulator $\mathrm{acc}_{V \cup \{i\}}$ and state $\mathrm{state}_{U \cup \{i\}}$, together with a witness wit_i for i.
- $\mathsf{AccUpdate}(pk_A, V, \mathrm{state}_U)$: outputs an accumulator acc_V for values $V \subset U$.
- $\mathsf{AccWitUpdate}(pk_A, wit_i, V_\omega, \mathrm{acc}_V, V, \mathrm{state}_U)$: outputs a witness wit_i' for acc_V if wit_i was a witness for acc_{V_ω} and $i \in V$.
- $\mathsf{AccVerify}(pk_A, i, wit_i, \mathrm{acc}_V)$: verifies that $v \in V$ uses an up-to-date witness wit_i and the accumulator acc_V. In that case the algorithm accepts, otherwise it rejects.

As Camenisch et al. [10] point out, the purpose of an accumulator is to have accumulator and witnesses of size independent of the number of accumulated elements.

3.7 Pedersen commitments

Using a commitment scheme, users can bind themselves to a chosen value without revealing the actual value to the third party receiving the commitment. Thereby, a user cannot change their choice (binding), and, at the same time, the recipient of a commitment does not learn anything about the actual value it contains (hiding of the value). Pedersen commitments [38] have public parameters as a group $\mathbb{G}$ of prime order q and generators $(g_0, \ldots, g_m)$. For committing to the value $(z_1, \ldots, z_m) \in Z_q$, a user picks a random $r \in Z_q$ and sets $C = \mathsf{PedCom}(z_1, \ldots, z_m;\ r) = g_0^r \prod_{i=1}^m g_i^{z_i}$.

4 DAMFA: DECENTRALIZED ANONYMOUS MULTI-FACTOR AUTHENTICATION

4.1 System Model

As shown in Fig. 5, the DAMFA protocol executes between four participants:

- U: The user who wants to access various services presented by different service providers. During the registration phase, U obtains a biometric template Bio and also a password pw from a sensor (which runs once). In the authentication phase, users need to interact with personal identity agents to authenticate themselves in an anonymous manner.

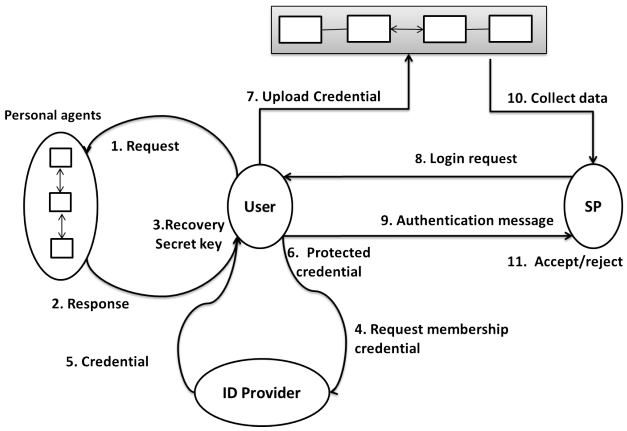

Figure 5: A system model of the DAMFA **scheme**

- PA_i (personal identity agents): We associate each user with some personal agents which act as auxiliary devices and assist a user in creating a credential for authentication. These personal agents remain under the administrative control of their associated user who can freely choose where to run them. For example, she can run them on a smart home controller in her residence, at a cloud provider, or even on her mobile phone.
- SP (verifiers): The service providers (untrusted and distributed servers) that request for authentication from U. After verifying user credentials, they provide the corresponding services.
- Nodes in the blockchain network: a number of users who enforce a specific credential issuing policy and collaboratively maintain a list of credentials in the ledger.
- IDP (Identity Provider): The identity provider known as a company or any other organization that provides different services to their employees or clients. IDP issues a credential to a client who intends to prove that it is a member of the legitimate group of this organization.

4.2 Threat Model

In order to demonstrate the security of the proposed protocol, we determine the capabilities and possible actions of an attacker. We consider a probabilistic polynomial-time attacker who has perfect control of the communication channels. He can eavesdrop all messages in public channels, and also modify, remove, and add messages on the network. The attacker can, at any time, corrupt $(t-1)$ of the user's agents (no more than threshold t), in which case the attacker knows all the long-term secrets (such as private keys or master shared keys).

Security and privacy requirements. In the proposed protocol, we consider some privacy requirements such as unlinkability, identity privacy, and user data privacy: Unlinkability means that an adversary cannot distinguish a user who is authenticating from any user who has authenticated in the past. Identity privacy means that an adversary cannot determine if a given authentication credential belongs to a specific user. User data privacy means that an adversary cannot learn anything about the user's sensitive authentication data (i.e., biometric data and password).

4.3 High-level view of our solution

To build a fully decentralized authentication architecture, we need to setup a small distributed shared database (to store credentials) between nodes. Data is highly available, but nobody has control over the database. Furthermore, users would never want to modify data in the past, user data needs to be immutable, and data should be publicly accessible. We employ a public append-only ledger in order to realize our requirements. A ledger maintains the integrity of the dataset and guarantees a consistent view of the ledger for every party. Every participant can append information to the ledger and once data is uploaded, nobody can delete or modify it which is exactly what we need.

Definition 4.1. A DAMFA system consists of a global transaction ledger instead of a single party representing the organization. Moreover, the DAMFA scheme consists of the following phases:

- Setup: In the setup phase, we define the public parameters and execute the following algorithm: U generates a private key and executes the Threshold Secret Sharing (TSS) on the private key to generate secret shares. The user stores secret shares among the personal agents. This step is similar to the initialization of the TOPRF scheme [27] which is done via a distributed key generation for discrete-log-based systems, e.g. [19].
- Registration: In the registration phase, the user U first selects a password pw and collects her biometric Bio at the sensor. Then, U runs TOPRF protocol by interacting with personal agents to reconstruct the TOPRF secret key. After that, IDP issues a membership credential that shows that U is a valid member (e.g. employee, account holder, subscribed user, etc.) For this propose, U sends a request with a pseudonym and a (non-interactive) zero-knowledge proof (NIZK) which indicates she is the owner of the pseudonym (she knows the secret key belong to the pseudonym) and authenticates herself to IDP. Then, U receives a membership credential which is a signature on her pseudonym.
  The user U creates a pseudonym Nym_u^o and verification information, namely a Protected-Credential PC_i, by encrypting the membership credential with TOPRF secret key. Subsequently, U computes a NIZK proof that (1) the credential PC_i and the pseudonym contain the same secret key and (2) proof of knowledge of the signature which is issued by the ID provider (i.e. she has valid group membership). Note that the user can execute these actions in an offline state because it is not required to interact with the public ledger. Finally, nodes should accept credentials to the ledger if and only if this proof is valid.
- Authentication: The user U attempts to access the services of a SP in an anonymous and unlinkable way. SP authenticates the user if and only if the user provides a valid credential. First, a service provider sends an authentication request which is a signature to U. The user inserts the password pw^* and the biometric Bio^* and runs TOPRF protocol by interacting with personal agents to reconstruct the TOPRF secret

value. U first scans the public ledger to obtain the accumulator AC, which is a set $\vec{\mathsf{PC}} = \{\mathsf{PC}_1, \ldots, \mathsf{PC}_n\}$ consisting of all credentials belonging to a specific IDP. Then, U finds her protected-credential PC_i^* from this set (via the her pseudonym Nym_u^o). U decrypts PC_i^* using the TOPRF secret key and recovers the initial credential (a signature from IDP). U presents the credential under a different pseudonym Nym_u^v by proving in zero-knowledge that (1) she knows a credential PC_i on the ledger from IDP, (2) the credential opens to the same secret key as her pseudonym Nym_u^v, and (3) she proves possession of a membership credential from IDP (the signature), cf. [17]. SP scans the public ledger to obtain the accumulator AC which is a set $\vec{\mathsf{PC}} = \{\mathsf{PC}_1, \ldots, \mathsf{PC}_n\}$ consisting of all credentials belonging to a specific organization. Then, it checks the validity of the candidate credential by finding the candidate credential in the set $\mathsf{PC}_i^* \in \vec{\mathsf{PC}}$ and checking a proof of knowledge on the credential and pseudonym.

4.4 Our Construction

4.4.1 Setup phases. We select a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ that is efficiently computable, non-degenerate, and all three groups have prime order p. We let g_1 and g_2 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, and $g_t = e(g_1, g_2)$ the generator of $\mathbb{G}_T$. Note that it is assumed to support one-way Bio-hash function $H_1()$ which resolves the recognition error of general hash functions [31]. We consider two additional hash functions as $H_2 : M \rightarrow \{0,1\}^\lambda$ and $H_g : M \rightarrow \mathbb{G}_1$. We publish $params \leftarrow (\mathbb{G}_1, \mathbb{G}_2, g_1, g_2, p, h_{nym}, H_1, H_2, H_g)$ as the set of system parameters where $h_{nym} \in \mathbb{G}_1$. The user U generates a private key K, then executes a secret sharing construction scheme on K to create the secret keys for each personal agent $\langle k_1, k_2, \ldots, k_n \rangle \leftarrow \mathsf{TSS}(K)$. U stores secret shares among the personal agents.

4.4.2 Registration Phase. To register a user to the system, U first chooses a password pw and scans her biometric impression Bio at the sensor. Then, U runs the following steps to register herself in the system.

- A user runs TOPRF protocol [27] with agents to compute the secret value $usk = F_K(\mathsf{pw}, \mathsf{Bio})$ as follows:
  - The user U picks a random number $r \in Z_p$ and computes $A = H_g(\mathsf{pw}, H(\mathsf{Bio}))^r$ and sends the message $M_1 = \{A\}$ to all PA_i.
  - Upon receiving the message $M_1 = \{A\}$ from the user, each PA_i computes $b_i = A^{k_i} = H_g(\mathsf{pw}, H_1(\mathsf{Bio}^*))^{\lambda_i \cdot k_i \cdot r}$ by Lagrange interpolation coefficients and secret key k_i (s.t. $K = \sum_{i \in \mathsf{SR}} \lambda_i \cdot k_i$). They return the message $M_2 = \{b_i\}$ to U.
  - After receiving all the messages b_i from personal agents, U computes: $C = \prod_{i \in \mathsf{SR}} b_i^{r^{-1}} = H_g(\mathsf{pw}, H_1(\mathsf{Bio}))^K \rightarrow usk = h(\mathsf{pw}, C)$.
- In order to obtain a membership credential from IDP, we use PS signatures protocol [39] to derive a signature on a hidden committed message as follows:
  - $\mathsf{KeyGen}(pp)$: The IDP runs this algorithm to generate private and public keys. This algorithm selects $(x, y, y_1) \leftarrow Z_p$, computes $(X, Y, Y_1) \rightarrow (g_1^x, g_1^y, g_1^{y_1})$ and $(X', Y', Y_1') \rightarrow (g_2^x, g_2^y, g_2^{y_1})$, and sets $sk \rightarrow (X, y, y_1)$ and $pk \rightarrow (g_1, g_2, Y, X', Y')$.
  - Protocol. A user first selects a random $r_2 \leftarrow Z_p$ and computes $C = g_1^{r_2} \cdot Y^{usk}$, which is a commitment on her secret key. She then sends C to the IDP. They both run a proof of knowledge of the opening of the commitment (authentication). If the signer is convinced, the IDP selects a random $u \leftarrow Z_p$ and returns $\sigma \leftarrow (\sigma_1 = g_1^u, \sigma_2 = (X \cdot C \cdot Y_1^m)^u)$. The user can now unblind the signature σ and get a valid signature over her secret key and the message m_1 by computing $\sigma \leftarrow (\sigma_1, \sigma_2/(\sigma_1)^{r_2})$ described in Sect. 3.1.
  - Verify. To verify this signature, the user can execute this algorithm and compute:
    $\mathsf{Verify}(pk, m, \sigma)$: $e(\sigma_1, X' \cdot Y'^{usk} \cdot Y_1'^{m_1}) = e(\sigma_2, g_2)$.
- CreatePC. The user generates a protected credential with TOPRF secret key usk derived from the password and the biometric: U picks a random number $s \in Z_p$ to generate a pseudonym as $\mathsf{Nym}_u^o = g_1^s \cdot h_{nym}^{usk}$ and computes an El-Gamal encryption of the credential σ with secret TOPRF values usk into a ciphertext as: $\mathsf{PC}_i = [\sigma]_{usk}$.
- Proof. A NIZK proof of knowledge of the credential (PS signature [39]) as follows: U selects random $r_3, t_1 \leftarrow Z_p$ and computes $\sigma' \leftarrow (\sigma_1^{r_3}, (\sigma_2 \cdot \sigma_1^{t_1})^{r_3})$. U sends $\sigma' = (\sigma_1', \sigma_2')$ to the verifier and carries out a zero-knowledge proof of knowledge (such as the Schnorr's interactive protocol) of m, usk and t_1 such that:

$$\pi = \mathsf{NIZK}\left\{ \begin{array}{l} (s, m_1, t_1, usk) : \mathsf{Nym}_u^o = g_1^s \cdot h_{nym}^{usk} \wedge \mathsf{PC}_i = Enc_{usk}(\sigma) \\ \\ \wedge\ e(\sigma_1', Y)^{usk} \cdot e(\sigma_1', g_2)^{t_1} \cdot (\sigma_1', Y_1)^{m_1} = \dfrac{e(\sigma_2', g_2)}{e(\sigma_1', X)} \end{array} \right\}.$$

- At the end of this phase, U submits the resulting values $(\mathsf{PC}_i, \pi, \mathsf{Nym}_u^o)$ to the public ledger nodes where π is a proof of knowledge on the Nym_u^o and the PC_i. If the signature verifies successfully, output 1, otherwise 0. The nodes should accept values to the ledger if this algorithm returns 1.

4.4.3 Authentication Phase. To this phase, a user authenticates herself to the service provider and establishes a session key with the service provider. The following steps are executed by U, PA_i, and SP:

- First of all, the server chooses a secret key $y \leftarrow Z_p$ and computes $Z \leftarrow g_1^y$. Then, SP generates a signature σ_s on message Z (i.e. Schnorr's signature [43]) using its secret key and sends the message $M_1 = \{Z, \sigma_s\}$ to the user.
- When receiving a pair (Z, σ_s), the client verifies whether σ_s is valid on message Z under the SP's public key. If σ_s is valid, U inserts pw^* and scans her personal biometric impression Bio^* at the sensor.
- The user interacts with personal agents and runs the necessary steps to compute the TOPRF protocol $F_K(\mathsf{Bio}^*, \mathsf{pw}^*) = usk = h(\mathsf{pw}^*, \prod_{i \in \mathsf{SR}} b_i^{r^{-1}})$. Then, U decrypts ciphertext $[\sigma]_{usk}$ with the TOPRF secret key usk to recover the credential σ.
- Show. The user creates a NIZK π ensuring that the credential is well-formed and the credential related to the same secret values as her pseudonym. Here we prove: (1) she knows a

credential on the ledger from the IDP, (2) the credential includes the secret key as her pseudonym, (3) she possesses of a credential (signature). We use the bilinear maps accumulator [10] to accumulate the group elements $\{g_1, \ldots, g_n\}$ instead of, e.g., the integers $\{1, \ldots, n\}$. In addition, Camenisch et al. [10] describe an efficient zero-knowledge proof of knowledge such as Schnorr's protocol [14, 43] that a committed value is in an accumulator. See [10, 12] to find how this proof works.

U runs the following steps to authenticate herself:

- The user selects a random number $r_4 \in Z_p$ to generate a pseudonym $\mathsf{Nym}_u^v = g_1^{r_4} \cdot h_{nym}^{usk}$ for communication with service providers.
- U picks random numbers $d, t_2 \leftarrow Z_p$ and computes a randomized commitment credential (like in the previous step) as $\sigma' \leftarrow (\sigma_1^{r_2}, (\sigma_2 \cdot \sigma_1^{t_2})^{r_2})$.
- Then, U calculates $D = g_1^d$ and a secret session key $SK = Z^d = g_1^{y \cdot d}$ and $\mathsf{Hmac}(SK, D, Z)$.
- For a set of credentials $\vec{\mathsf{PC}}$, U computes an accumulator and witness as $\mathsf{AC} = \mathsf{Accumulate}(\mathsf{params}, \vec{\mathsf{PC}})$ and $\omega = \mathsf{GenWitness}(\mathsf{params}, \vec{\mathsf{PC}}, \mathsf{PC}_i^*)$, carries out a zero-knowledge proof of knowledge of the credential, and outputs the following proof of knowledge π such that:

$$\mathsf{NIZK}\left\{\begin{array}{l}(usk, \omega, d, t_2, m, r_4) : \mathsf{AccVerify}(\mathsf{params}, \mathsf{AC}, \omega) = 1 \\ \wedge\, e(\sigma_1', Y)^{usk} \cdot e(\sigma_1', g_2)^{t_2} \cdot (\sigma_1', Y)^m = \dfrac{e(\sigma_2', g_2)}{e(\sigma_1', X)} \\ \wedge\, \mathsf{PC}_i = Enc_{usk}(\sigma) \wedge D = g_1^d \wedge \mathsf{Nym}_u^v = g_1^{r_4} \cdot h_{nym}^{usk}\end{array}\right\}.$$

Finally, U sends the message $M_4 = \{\mathsf{Nym}_u^v, D, \mathsf{Hmac}, \pi\}$ to the service provider.

- After receiving the message $M_4 = \{\mathsf{Nym}_u^v, D, \mathsf{Hmac}, \pi\}$ from the user, the service provider first scans through the ledger to obtain a set $\vec{\mathsf{PC}}$ consisting of all credentials belonging to IDP. First, SP computes the accumulator $AC = \mathsf{Accumulate}(\mathsf{params}, \vec{\mathsf{PC}})$. Then, it verifies that $\pi = 1$ is the aforementioned proof of knowledge on PC_i and Nym_u^v using the known public values. If the proof verifies successfully, output 1, SP computes the session key as follows: $SK = D^y = g_1^{y \cdot d}$.

Then, SP computes $\mathsf{Hmac}^*(SK, D, Z)$ and checks $\mathsf{Hmac} = \mathsf{Hmac}^*$. If $\pi = 1$ and Hmac holds, SP accepts SK as the session key and also the user is authentic.

Note that we can simply send σ' alongside the message of the proof of knowledge. With this, we can prove the construction is a $\sum$-protocol (see [39] to see how proof of knowledge of PS signature works).

Theorem 4.2. *Our proposed protocol is secure against any non-uniform* PPT *adversary corrupting* $t - 1$ *many personal agents* $\mathsf{PA_i}$ *by assuming that the El-Gamal encryption, zero knowledge proof of signature, and the* TOPRF *function are secure and also the hash function is collision resistant.*

Proof sketch. Our construction DAMFA is modular, and relies directly on the TOPRF and the zero-knowledge proof we associate with it for the authentication algorithm. The security is then straightforwardly inherited from those algorithms:

Credential security requires that no adversary is able to present a credential (guess password and biometric) and generate a session key, which they should not have had any access to. If we use a TOPRF on passwords and biometric of users, then security properties of TOPRF would make it hard to guess them. The proof is once again two-fold:

- First, the authentication is done through a zero-knowledge proof. At this step, either the adversary presents an invalid credential, or manages to build a valid proof. Hence, the adversary breaks the soundness of the underlying proof of knowledge we used, or otherwise, uses a valid credential.
- At this step, we now assume the adversary wins by using a valid credential. We now rely on the obliviousness of the TOPRF. We interact with a TOPRF challenge, to answer every adversarial request, and at the end, we can use the (valid) credential output by the adversary to break the TOPRF obliviousness, which leads to the conclusion.

Anonymity. During the registration phase, when a user reveals her pseudonym, but does not (intentionally) reveal her secret key *usk*, no adversary should learn any information about the secret key and the identity. In addition, during the authentication phase, a user proves her credential using a zero knowledge proof which reveals no additional information about her secret key and identity to the SP.

5 IMPLEMENTATION

In this section, we illustrate the practicability of the proposed protocol. To this end, we provide the public ledger part which is realized by a well-known blockchain, namely Namecoin. The results are summarized in Table 1.

5.1 Namecoin implemention

The public ledger can be implemented by a blockchain system. One of the smooth ways to realize a public ledger is using Namecoin blockchain. Namecoin allows registering names and stores related values in the blockchain which is a securely distributed shared database. It also enables a basic feature to query the database and to retrieve the list of existing names and associated data. Thus, we can store credentials, scan them based on namespace and then verify them. We execute the following steps in order to participate in the Namecoin system and store credentials by the namecoin *id* as pseudonyms:

- We need to install a Namecoin client that has a full copy of the Namecoin blockchain and keep it in sync with the P2P network by fetching and validating new blocks from connected peers. We use the implementation of Namecoin client

Table 1: Public Ledger Instantiations

Properties	Namecoin
Initial Data Size	≈ 5.08 GB
Initial Sync Time	≈ 3h
Cost	0.069 USD
Confirmation Time	10min/2h

[47], which can be controlled by HTTP JSON-RPC, command line, or graphical interface. It spontaneously connects to the Namecoin network and downloads the blockchain.

- The Namecoin client also creates the user's wallet which includes the private key of Namecoin addresses of the user.
- To save credentials in the blockchain, the user needs to register a namespace "`id/name`" as the owner of the name by paying a very small fee (currently 0.0064 USD). An *id* name can be registered using the Namecoin graphical interface or commands "name_new" and "name_firstupdate". The following description shows how the *id* name in Namecoin namespace is registered and how those names can be accessed.

  ```
  namecoind name-new id/3608a30756b07e13cf58cfd5...
  ```

  The output will look like this:

  ```
  [ "0e0e03510b0b0b7dbba6e301e519693f68062121b29f3
    cd3a6652c238360d0d0", "9f213ff4a582fd65" ]
  ```

  This transaction shows a hashed version of the name, salted with a random value (in this example is "9f213..." for transaction ID "0e0e0351...").
- The user can store arbitrary data (contains a credential (description)) for Namecoin keys using JSON format: the following codes can be a simple example of the JSON value of an identity name:

  ```
  namecoind name_firstupdate id/3608a30756b07e13...
  ```

  Output:

  ```
  { "description" : "28790de641755e77d13382229156
    f5c26a9dd8a9673006b154d6c8a707e7...",
    "namecoin" : "NBvmSUQbRGu..." }
  ```
- Subsequently, the update has been confirmed and transactions have been added to the blockchain. The user has a fully valid credential. To show the credential, SP scans through the list of added names and retrieves all credentials via a graphical interface or like the following code:

  ```
  namecoind name_list
  ```

  Output:

  ```
  [ { "name" : "id/3608a30756b07e13cf58cfd5...",
      "value" : "28790de641755e77d13382229156f5c2
      6a9dd8a9673006b154d6c8a707e7...",
      "address" : "NBvmSUQbRGunCSoxA3dE22q2sq...",
      "expires_in" : 36000 } ]
  ```

Cost. Initially, a reasonable transaction fee of either 0.00 or 0.01 NMC is charged. We can choose this fee based on how fast we want to process a transaction.

Latency. Namecoin and Bitcoin both attempt to generate blocks every 10 minutes; on average, it takes nearly 5 minutes to see the data appear on the blockchain. In practice, it then takes the necessary time to solidify the transactions and the data to be verified. For Namecoin, it takes about 2 hours to confirm that the data are uploaded in the blockchain (12 confirmations). That is why name-firstupdate will only be accepted after a mandatory wait period of 12 additional blocks.

Remark 1. *Note that these costs and delays occur only once during the setup and registration phases. They do not affect the authentication phase. Thus, we focus on the computation time of the authentication phase that is used frequently in the authentication system (see section 5.2).*

5.2 Performance of the Authentication System

We now examine the performance of our decentralized anonymous authentication system. There are two main steps: the registration phase and the authentication phase. First, we measure the performance of the registration protocol: The time for a user to generate a pseudonym and corresponding credential. In the registration phase, the service providers do not compute anything. Second, we test the performance of the authentication protocol including the time for generating a candidate credential and pseudonym, the time for a personal agent to compute a TOPRF response, and also the time for verifying a proof by running bilinear pairing operation by the service provider. To simplify the criterion for evaluating the experiment result, we only assume a simple policy with a threshold $t = 2$ for two agents (increasing t will effect PC and AA sub protocols). The experiment is conducted on a laptop with Intel Core i5-6200U CPU 2.30GHz, 8.00 GB RAM, and 64-bit Ubuntu OS. The test code is written in Java based on BouncyCastle and PBC libraries (version 1.0.2) and TSS [44]. To generate the curve parameters, we use pairings: $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow G_T$ constructed on the curve $y^2 = x^3 + x$ over the field F_q for some prime $q = 3 \mod 4$. Both G_1 and G_2 are groups of points $E(F_q)$. The prime order r is 160 bits and q is a 512-bit prime number. We use the hash function SHA256. Table 2 shows the computational performance of the authentication and the registration phases over 100 trials. In the registration phase, we denote GOC, GC-P, and PC to define time consumed in generating only a credential and a pseudonym, time executed in generating a credential and a pseudonym with pre-computations (TSS and key generation), and time consumed in pre-computations (PC), respectively. In the authentication phase, we denote VC, CC, AA and Total to define time consumed in verifying the validity of a credential (service provider side), generating a candidate credential (user side), agent time in the authentication phase, and total time in the authentication phase, respectively.

Table 2: The performance of the registration protocol and the authentication protocol

Registration Phase		Authentication Phase	
Sub protocol	Time	Sub protocol	Time
GOC	≈ 82 ms	VC	≈ 555 ms
GC-P	≈ 663 ms	CC	≈ 60 ms
PC	≈ 541 ms	AA	≈ 6 ms
Total	≈ 703 ms	Total	≈ 640 ms

6 CONCLUSION

In this paper, we proposed a decentralized authentication and key exchange system DAMFA under TOPRF protocol and standard cryptographic primitives. The proposed scheme builds upon a trustworthy global append-only ledger that does not rely on a trusted server. DAMFA fulfills the following properties:

(1) *Decentralization property* means the process of user authentication no longer depends on a trusted third party. To realize such a distributed ledger, we propose using the blockchain system already in real-world use with the cryptographic currency Bitcoin.

(2) *Passive verification* means that service providers with access to the shared ledger can verify users and that a given user has registered a particular credential without requiring interaction with an identity provider.

(3) *Single sign-on property* ensures that a user logs in with a single ID into the identity provider and then gains access to any of several related systems. So, users do not need to register with each service provider individually.

(4) *Anonymity* guarantees that no one can trace or learn information about the user's identity during the authentication process. Finally, we evaluated that our protocol is efficient and practical for authentication systems.

ACKNOWLEDGMENTS

This work has been carried out within the scope of the LIT Secure and Correct Systems Lab funded by the State of Upper Austria and has partially been supported by Digidow, the Christian Doppler Laboratory for Private Digital Authentication in the Physical World, funded by the Christian Doppler Forschungsgesellschaft, 3 Banken IT GmbH, Kepler Universitätsklinikum GmbH, NXP Semiconductors Austria GmbH, and Österreichische Staatsdruckerei GmbH.

REFERENCES

[1] Shashank Agrawal, Peihan Miao, Payman Mohassel, and Pratyay Mukherjee. 2018. PASTA: PASsword-based Threshold Authentication. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2042–2059.

[2] Gergely Alpár, Jaap-Henk Hoepman, and Johanneke Siljee. 2011. The identity crisis. security, privacy and usability issues in identity management. *arXiv preprint arXiv:1101.0427* (2011).

[3] Auth0 Inc. 2019. JSON Web Tokens. https://jwt.io/

[4] Mihir Bellare, David Pointcheval, and Phillip Rogaway. 2000. Authenticated Key Exchange Secure Against Dictionary Attacks. In *Proceedings of the International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT)*. Springer, 139–155.

[5] Steven M. Bellovin and Michael Merritt. 1992. Encrypted Key Exchange: Password-Based Protocols SecureAgainst Dictionary Attacks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE.

[6] Steven M. Bellovin and Michael Merritt. 1993. Augmented Encrypted Key Exchange: A Password-based Protocol Secure Against Dictionary Attacks and Password File Compromise. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. ACM, 244–250. https://doi.org/10.1145/168588.168618

[7] Fabrice Benhamouda and David Pointcheval. 2013. Verifier-Based Password-Authenticated Key Exchange: New Models and Constructions. *IACR Cryptology ePrint Archive* 2013 (2013), 833.

[8] Victor Boyko, Philip MacKenzie, and Sarvar Patel. 2000. Provably Secure Password-authenticated Key Exchange Using Diffie-Hellman. In *Proceedings of the International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT)*. Springer, 156–171.

[9] Julien Bringer, Hervé Chabanne, and Roch Lescuyer. 2016. Software-Only Two-Factor Authentication Secure Against Active Servers. In *International Conference on Cryptology in Africa (AFRICACRYPT)*. Springer, 285–303. https://doi.org/10.1007/978-3-319-31517-1_15

[10] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. 2009. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *International Workshop on Public Key Cryptography*. Springer, 481.

[11] Dell Cameron. 2017. Over 560 Million Passwords Discovered in Anonymous Online Database (2017). https://goo.gl/upDqzt

[12] Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. 2014. Algebraic MACs and Keyed-Verification Anonymous Credentials. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 1205–1216. https://doi.org/10.1145/2660267.2660328

[13] Daniel Fett, Ralf Küsters, and Guido Schmitz. 2017. The web SSO standard openid connect: In-depth formal security analysis and security guidelines. In *Computer Security Foundations Symposium (CSF)*. IEEE, 189–202.

[14] Amos Fiat and Adi Shamir. 1986. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the Theory and Application of Cryptographic Techniques (Eurocrypt)*. Springer, 186–194.

[15] Nils Fleischhacker, Mark Manulis, and Amir Azodi. 2014. A modular framework for multi-factor authentication and key exchange. In *International Conference on Research in Security Standardisation*. Springer, 190–214. https://doi.org/10.1007/978-3-319-14054-4_12

[16] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. 2014. A Decentralized Public Key Infrastructure with Identity Retention. *IACR Cryptology ePrint Archive* (2014). https://eprint.iacr.org/2014/803.pdf

[17] Christina Garman, Matthew Green, and Ian Miers. 2014. Decentralized Anonymous Credentials. In *The Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[18] Gemalto. 2015. 2014 Year of mega breaches & identity theft: Findings from the breach level index. https://www.sbs.ox.ac.uk/cybersecurity-capacity/content/2014-year-mega-breaches-and-identity-theft

[19] Rosario Gennaro, Stanisław Jarecki, Hugo Krawczyk, and Tal Rabin. 1999. Secure distributed key generation for discrete-log based cryptosystems. In *International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt)*. Springer, 295–310.

[20] Rosario Gennaro and Yehuda Lindell. 2003. A framework for password-based authenticated key exchange. In *International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt)*. Springer, 524–543.

[21] Vindu Goel and Nicole Perlroth. 2016. Yahoo Says 1 Billion User Accounts Were Hacked. https://goo.gl/q4WZi9

[22] Oded Goldreich and Yehuda Lindell. 2006. Session-key generation using human passwords only. *Journal of Cryptology* 19, 3 (2006), 241–340.

[23] Adam Groce and Jonathan Katz. 2010. A New Framework for Efficient Password-based Authenticated Key Exchange. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. ACM, 516–525.

[24] Dick Hardt and Michael Jones. 2012. The oauth 2.0 authorization framework: Bearer token usage. (2012).

[25] Debiao He and Ding Wang. 2015. Robust biometrics-based authentication scheme for multiserver environment. *IEEE Systems Journal* 9, 3 (2015), 816–823.

[26] Xinyi Huang, Yang Xiang, Elisa Bertino, Jianying Zhou, and Li Xu. 2014. Robust multi-factor authentication for fragile communications. *IEEE Transactions on Dependable and Secure Computing* 11, 6 (2014), 568–581.

[27] Stanisław Jarecki, Aggelos Kiayias, Hugo Krawczyk, and Jiayu Xu. 2017. TOPPSS: Cost-minimal Password-Protected Secret Sharing based on Threshold OPRF. In *International Conference on Applied Cryptography and Network Security*. Springer, 39–58. https://doi.org/10.1007/978-3-319-61204-1_3

[28] Stanislaw Jarecki, Hugo Krawczyk, Maliheh Shirvanian, and Nitesh Saxena. 2016. Device-Enhanced Password Protocols with Optimal Online-Offline Protection. In *Proceedings of the ACM on Asia Conference on Computer and Communications Security (ASIA CCS)*. ACM, 177–188. https://doi.org/10.1145/2897845.2897880

[29] Stanislaw Jarecki, Hugo Krawczyk, Maliheh Shirvanian, and Nitesh Saxena. 2018. Two-Factor Authentication with End-to-End Password Security. In *International Conference on Practice and Theory of Public Key Cryptography (PKC)*. Springer, Cham. https://doi.org/10.1007/978-3-319-76581-5_15

[30] Stanislaw Jarecki and Xiaomin Liu. 2009. Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. In *Proceedings of the Conference on Theory of Cryptography (TCC)*. Springer, 577–594. https://doi.org/10.1007/978-3-642-00457-5_34

[31] Andrew Teoh Beng Jin, David Ngo Chek Ling, and Alwyn Goh. 2004. Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern recognition* 37, 11 (Nov. 2004), 2245–2255. https://doi.org/10.1016/j.patcog.2004.04.011

[32] Yehuda Lindell. 2011. Anonymous authentication. *Journal of Privacy and Confidentiality* 2, 2 (2011).

[33] Omid Mir and Morteza Nikooghadam. 2015. A secure biometrics based authentication with key agreement scheme in telemedicine networks for e-health services. *Wireless Personal Communications* 83, 4 (2015), 2439–2461.

[34] Omid Mir, Theo van der Weide, and Cheng-Chi Lee. 2015. A secure user anonymity and authentication scheme using AVISPA for telecare medical information systems. *Journal of medical systems* 39, 9 (2015), 89.

[35] Aleksandr Ometov, Sergey Bezzateev, Niko Mäkitalo, Sergey Andreev, Tommi Mikkonen, and Yevgeni Koucheryavy. 2018. Multi-factor authentication: A survey. *Cryptography* 2, 1 (2018), 1.

[36] OneLogin, Inc. 2019. SAML Toolkits. https://developers.onelogin.com/saml
[37] Ian Paul. 2012. LinkedIn Confirms Account Passwords Hacked. http://goo.gl/UBWuY0
[38] Torben Pryds Pedersen. 1991. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual International Cryptology Conference (Crypto)*. Springer, 129–140.
[39] David Pointcheval and Olivier Sanders. 2016. Short randomizable signatures. In *Cryptographers' Track at the RSA Conference*. Springer, 111–126.
[40] David Pointcheval and Sébastien Zimmer. 2008. Multi-factor Authenticated Key Exchange. In *6th International Conference on Applied Cryptography and Network Security (ACNS)*. Springer, 277–295.
[41] David Recordon and Drummond Reed. 2006. OpenID 2.0: A Platform for User-centric Identity Management. In *Proceedings of the Second ACM Workshop on Digital Identity Management (DIM)*. ACM, 11–16. https://doi.org/10.1145/1179529.1179532
[42] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE, 459–474. https://doi.org/10.1109/SP.2014.36
[43] Claus-Peter Schnorr. 1990. Efficient Identification and Signatures for Smart Cards. In *Proceedings of the Annual International Cryptology Conference on Advances in Cryptology (CRYPTO)*. Springer, 239–252.
[44] Adi Shamir. 1979. How to share a secret. *Commun. ACM* 22, 11 (1979), 612–613.
[45] SeongHan Shin, Kazukuni Kobara, and Hideki Imai. 2010. Anonymous password-authenticated key exchange: New construction and its extensions. *IEICE transactions on fundamentals of electronics, communications and computer sciences* 93, 1 (2010), 102–115.
[46] Maliheh Shirvanian, Stanislaw Jarecki, Nitesh Saxena, and Naveen Nathan. 2014. Two-Factor Authentication Resilient to Server Compromise Using Mix-Bandwidth Devices. In *Network and Distributed System Security Symposium (NDSS)*.
[47] Namecoin team. 2016. Namecoind, sourcecode of the Namecoin-client reference implementation. https://github.com/namecoin/namecoin
[48] Duong Quang Viet, Akihiro Yamamura, and Hidema Tanaka. 2005. Anonymous Password-based Authenticated Key Exchange. In *Proceedings of the International Conference on Cryptology in India (INDOCRYPT)*. Springer, 244–257. https://doi.org/10.1007/11596219_20
[49] Ding Wang, Haibo Cheng, Ping Wang, Xinyi Huang, and Gaopeng Jian. 2017. Zipf's law in passwords. *IEEE Transactions on Information Forensics and Security* 12, 11 (2017), 2776–2791. https://doi.org/10.1109/TIFS.2017.2721359
[50] Jing Yang and Zhenfeng Zhang. 2008. A new anonymous password-based authenticated key exchange protocol. In *International Conference on Cryptology in India (INDOCRYPT)*. Springer, 200–212. https://doi.org/10.1007/978-3-540-89754-5_16
[51] Rupeng Yang, Man Ho Au, Qiuliang Xu, and Zuoxia Yu. 2019. Decentralized blacklistable anonymous credentials with reputation. *Computers & Security* 85 (2019), 353–371.
[52] Rui Zhang, Yuting Xiao, Shuzhou Sun, and Hui Ma. 2017. Efficient Multi-Factor Authenticated Key Exchange Scheme for Mobile Communications. *IEEE Transactions on Dependable and Secure Computing* (2017), 1.