Quantum-Resistant Password-Based Threshold Single-Sign-On Authentication with Updatable Server Private Key

Jingwei Jiang[1], Ding Wang[2,3](✉), Guoyin Zhang[1], Chenzhi Yuan[1]

[1] College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China
[2] College of Cyber Science, Nankai University, Tianjin 300350, China
[3] Tianjin Key Laboratory of Network and Data Security Technology, Nankai University, Tianjin 300350, China
wangding@nankai.edu.cn

Abstract. Passwords are the most prevalent authentication mechanism, and proliferate on nearly every new web service. As users are overloaded with the tasks of managing dozens even hundreds of passwords, accordingly password-based single-sign-on (SSO) schemes have been proposed. In password-based SSO schemes, the authentication server needs to maintain a sensitive password file, which is an attractive target for compromise and poses a single point of failure. Hence, the notion of password-based threshold authentication (PTA) system has been proposed. However, a static PTA system is threatened by perpetual leakage (e.g., the adversary perpetually compromises servers). In addition, most of the existing PTA schemes are built on the intractability of conventional hard problems, and become insecure in the quantum era.

In this work, we first propose a threshold oblivious pseudorandom function (TOPRF) to harden the password so that PTA schemes can resist offline password guessing attacks. Then, we employ the threshold homomorphic aggregate signature (THAS) over lattices to construct the first quantum-resistant password-based threshold single-sign-on authentication scheme with the updatable server private key. Our scheme resolves various issues arising from user corruption and server compromise, and it is formally proved secure against quantum adversaries. Comparison results show that our scheme is superior to its counterparts.

Keywords: Password; Single-Sign-On; Threshold Authentication; Oblivious Pseudorandom Function; Lattice.

1 Introduction

Password-based authentication is the most prevalent mechanism for validating users. Passwords have various advantages of being memorable, convenient, and regeneration, and it is unlikely to be replaced in the near future [1,2]. However, recent research reveals that each user has 80∼100 password accounts on average [3,4]. The management of such an amount of usernames and passwords for diverse applications is challenging, and would lead to insecure behaviors like reuse and writing down [5,6]. Single-sign-on (SSO) reduces this burden [7]. Specifically, password-based SSO verifies the username and password through a trusted identity server and generates an access token for the authenticated user [8]. The user can access *various* application servers within a specified period of time through the access token without showing them the password [9].

An inherent limitation of password-based SSO is that the server may become a single point of failure. When the server is compromised, an overwhelming fraction of users' passwords will be exposed, even if passwords are properly stored in salted-hash or memory-hard functions [10,11]. On the other hand, the adversary can obtain the master secret key and forge arbitrary tokens that enable access to arbitrary resources and information in the system [12].

A viable solution to this single point of failure is to employ threshold cryptography where a distributed protocol executes on N servers. This is a widely applied idea in threshold password-authenticated key exchange [13,14] and threshold password-authenticated secret sharing [15,16]. Agrawal et al. [12] proposed a framework of password-based threshold authentication (PTA) scheme, called PASTA, which employs the threshold oblivious pseudorandom function (TOPRF) [17] to ensure password security. However, its inability to actively update the server's private key makes it vulnerable to perpetual leakage. Categorically, the adversary will continuously try to compromise servers and collect more than threshold shares over a long period of time. Thus the adversary will forge authentication tokens or even take control of the entire system.

There are various dynamic PTA schemes to solve the perpetual leakage [18,19,20]. More concretely, Zhang et al. [18] use a key technique to require all identity servers to share 0 in a distributed way. Each server can renew the master secret share by adding the additional share of 0. Baum et al. [19] add a backup key to derive a new master private key based on using 0 sharing. Rawat et al. [20] extend the function of PASTA and add the function of password update. As we know, most of the existing PTA schemes are built on RSA-based or pairing-based cryptosystems. Unfortunately, with quantum computers' continuous development and progress [21,22], all those PTA schemes, which are based on traditionally intractable problems (e.g., large integer factorization problem and discrete logarithm problem), are vulnerable to quantum attacks.

In the study of the quantum-resistant schemes, lattice-based schemes are regarded as the most promising general-purpose algorithms for public-key encryption by NIST [23,24]. Many quantum-resistant password-only [25,26] and authenticated key exchange schemes [27,28] have been proposed over lattices. Inspired by these lattice-based schemes, we believe that the lattice-based scheme is a promising choice for PTA. However, to the best of our knowledge, there is a lack of mature components for constructing PTA schemes over lattices, including the TOPRF for resisting offline password guessing attacks and the threshold homomorphism aggregation signature (THAS) for building threshold token generation (TGG). Thus, the main goal of our solution is to answer the following questions:

Is it possible to design a quantum-resistant dynamic password-based threshold SSO authentication scheme with TOPRF and THAS over lattices?

Our answer to the above question is affirmative. Next, we overview the concrete results and necessary contributions, showing the high-level ideas and components used throughout quantum-resistance PTA schemes.

1.1 Motivation

The large-scale quantum computers can execute Shor's [29] and Grover's [30] algorithms, which will greatly threaten the traditional cryptography system. Group-based and pair-based cryptographic schemes will no longer be secure. The standards organizations (e.g., IEEE, IETF, and NIST) have begun to prepare for the collection of quantum-resistant algorithms. According to the plan of NIST [23,24], the standard of post-quantum cryptography will be available in 2022-2024. In the status report on the third round of the NIST post-quantum cryptography standardization process [31], the lattice-based algorithm is the most popular quantum-resistant algorithm (five of the seven algorithms officially selected for the third round belong to lattice cryptography). The advantages of lattice-based cryptography include: i) The worst-case to average-case reductions; ii) Both public key encryption and digital signatures are taken into consideration; iii) The ability to construct complex cryptographic primitives (e.g., fully homomorphic encryption). With these advantages, we believe that a quantum-resistant dynamic password-based threshold single-sign-on authentication scheme can be constructed on the lattice.

In addition, PASTA [12] as the round-optimal PTA scheme reduces the interaction to two flows on the network. If the user enters the correct username[4]/password, the user can recover the authentication token locally after the interaction. However, servers do not know the login result of the user, and the scheme is vulnerable to online dictionary attacks. (Although online password guessing attacks can be prevented by limiting the login times of a single user, it is unrealistic to restrict the login times of all users at the same time). A feasible scheme is to give up the round optimization and calculate the token information on servers to improve the security, such as the scheme of Zhang et al. [18].

Furthermore, to deal with potential threats in SSO, *the PTA scheme needs to use a series of essential components, including distributed key generation, TTG, TOPRF, and proactive secret renewal.* These components have been widely studied under the conventional hard problems and can be handily applied to PTA [12,18,19,20]. However, the lattice-based homomorphic group encryption provides no support for password re-randomization. In addition, threshold aggregation magnifies noise over lattices that may cause users to derive different authentication information based on a password. These mature research results are challenging to construct quantum-resistant PTA schemes directly. It is necessary to use or design lattice-based components.

1.2 Contributions

The above motivations prompt us to design a two-round password-based threshold single-sign-on authentication over lattices, consisting of registration, login

[4] For convenience, we write username as ID

and authentication, server private key renewal, and password update. To summarize, we make the following key contributions.

- We design a threshold homomorphism aggregation signature to construct a threshold token generation (TTG) protocol.To address the fact that no mature tool like the BLS [32] over lattices can directly construct TTG schemes, we use the universal thresholdizer [33] to add a threshold to a lattice-based aggregate signature, and propose an unforgeable TTG scheme over lattices.
- We improve the oblivious pseudorandom function (OPRF) of Albrecht et al. [34], and design a new threshold OPRF (TOPRF) over lattices for the first time. We show the threshold constraints for TOPRF to maintain correctness over lattices. Furthermore, we prove that unpredictability and obliviousness are satisfied between the password and the derived authentication information. It means that the adversary can neither obtain the corresponding password through the authentication information, nor predict the authentication information derived from any password.
- We propose the first password-based threshold single-sign-on authentication scheme over lattices, based on our TTG and TOPRF constructed above. Our scheme supports the feature that users update passwords and servers update private keys. No user assistance is required when the server updates the private key, and the generated authentication token is not invalidated after the private key is updated. We evaluate the security of our scheme under two parameter settings, and make a rigorous security proof based on the Bellare-Pointcheval-Rogaway (BPR)-like model [35] by the sequences of games. In addition, we show the computational overhead of each part in our scheme under different security levels. Comparison results show that our scheme is superior to its counterparts [9,12,18,19,20].

2 Preliminaries

Notions. For any integer q, there is a polynomial $R_q := \mathbb{Z}_q[X]/\langle X^n+1\rangle$ in the power-of-two cyclotomic ring. Let upper-case letter A and lower-case bold letter $\mathbf{x}$ denote matrices and vectors, respectively. $\lfloor\cdot\rceil$ indicates that a rational number is rounded to the nearest natural number. We use $x \xleftarrow{\$} \mathcal{D}$ to denote the sampling of x according to distribution $\mathcal{D}$. We write $x \xleftarrow{R} S$ to indicate sampling uniformly at random from a finite set S. The notation $\approx_c$ to denote computationally indistinguishable. Finally, κ denote the security parameter.

2.1 Lattices, SIS, and DRLWE

For an m-dimensional lattice $\Lambda = \{As | s \in \mathbb{Z}^n\}$ with $A \in \mathbb{Z}^{m\times n}$ for $m \geq n\lceil log_2 q\rceil$ and the Gaussian distribution as $exp(-\pi\frac{(x-c)^2}{\sigma^2})$, where c is a centre parameter, we call a distribution χ is (B,δ)-bounded if $Pr[x \xleftarrow{\$} \chi | |x| \geq B] \leq \delta$.

Definition 1 (SIS$_{q,m,n,\sigma}$). *Let $q, m, n, \sigma > 0$ depend on κ. The $SIS_{q,m,n,\sigma}$ problem is to find a nonzero vector $\mathbf{v} \in R_q$ of norm $\|\mathbf{v}\| \leq \delta$ such that $A\mathbf{v} = \mathbf{0} \in R_q$. More formally, for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, we define the advantage $Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa) = |Pr[\mathcal{A}(\mathbf{v} \leq \delta : A\mathbf{v} = \mathbf{0} \in R_q)]| \leq \varepsilon(\kappa)$.*

Definition 2 (DRLWE$_{q,m,n,\sigma}$, [36]). *Let $q, m, n, \sigma > 0$ depend on κ. The DRLWE$_{q,m,n,\sigma}$ problem is to distinguish between $(A, A \cdot s + e_i)_{i\in[m]} \in (R_q)^2$ and $(A, r_i)_{i\in[m]} \in (R_q)^2$ for $A, r_i \overset{R}{\leftarrow} R_q$; $s, e_i \overset{R}{\leftarrow} R(\chi_\sigma)$. For any PPT $\mathcal{A}$, we define that $Adv_{\mathcal{A}}^{\mathsf{DRLWE}}(\kappa) = |Pr[\mathcal{A}(q, m, n, \sigma, A, s)] - Pr[\mathcal{A}(q, m, n, \sigma, A, r_i)]| \leq \varepsilon(\kappa)$.*

2.2 Distributed Key Generation Protocol over Lattices

Since lattice is an infinite additive group, it can not be directly combined with Shamir secret sharing scheme [37]. Fortunately, we can share elements of a finite abelian quotient group $\mathbb{G}$ with identity element 0 by (t, N)-threshold secret sharing [38]. Let $e(\mathbb{G})$ denote exponent of $\mathbb{G}$ and $s \in \mathbb{G}$. We have that:

Definition 3. *There is the smallest $m \in \mathbb{Z}^+$ such that $ms = s + s + ... + s = 0$, i.e. s is a module over the ring $R = \mathbb{Z}_e(s)$. The value s can be share by a formal polynomial $f(X) = \sum_{j=0}^{t} f_j X^j \in S[X]$ of the maximum degree at $t - 1$, where $f(0) = s$ and the $f(i) \in \mathbb{G}$ for $i \in [1, N]$ are uniformly random and independent. At least t participants can reconstruct the secret s, there exists an efficiently computable Lagrange coefficients $\lambda_{i,j} \in \mathbb{Z}_q$ such that $s = \sum_{i=1}^{t} \lambda_{i,j} \cdot f(i)$.*

To use the above secret sharing over lattices, we also need to set relevant parameters. Let $k \geq \log_p(n + 1)$, where p is the smallest prime divisor of $e(\mathbb{G})$, we can share $s \in \mathbb{G}$ among n servers using shares in $\mathbb{G}^k$. By [38], we can use $R = \mathbb{Z}_e(\mathbb{G})[X]/F(X)$ for any monic degree-k polynomial $F(X) = \sum_{i=0}^{k} F_i X^i \in \mathbb{Z}_e(\mathbb{G})$ that is irreducible modulo every prime dividing $e(\mathbb{G})$ that is irreducible modulo every prime dividing $e(\mathbb{G})$. We write $[s]^i$ to denote i-th server's share and the tuple of all shares by $[s]$. By employing integer sampling and MPC, a distributed server key generation without the trusted center is as follows:

Definition 4. *The distributed key generation $\mathcal{F}_{KG}$ must contain the following two polynomial algorithms:*

- *$[s_i] \leftarrow \mathsf{Genshare}(\mathbb{Z}_e(\mathbb{G}), \mathbb{Z}_q)$ is a probabilistic algorithm that sample a series of polynomials $F_i \overset{\$}{\leftarrow} \mathbb{Z}_e(\mathbb{G})$ and generates a series of shares $[s_i] \overset{R}{\leftarrow} \mathbb{Z}_q$.*
- *$sk_j \leftarrow \mathsf{Genkey}(i, j, [s_i]^j)$ is a deterministic algorithm that generates secret key $sk_j = \sum_{i=1}^{N} [s_i]^j$ by receiving n tuple of $(i, j, [s_i]^j)$. After receiving n numbers of $[s_i]^j$, S_j computes $sk_j = \sum_{i=1}^{N} [s_i]^j$.*

An unknown master secret key $msk = \sum_{j=1}^{t} [s]_j^0$ that cannot be recovered unless at least t malicious servers collude.

2.3 Threshold Homomorphic Aggregate Signatures over Lattices

In a PTA scheme, servers generate an authentication token for the user (also called the client) by executing threshold token generation (TTG). A threshold aggregate signature protocol can construct a *Threshold Token Generation* (TTG) scheme. Lattice-based *Homomorphic Aggregate Signature* (HAS) is a variant of linear homomorphic signature [**?**]. Compared with the signature under the traditional cryptography system, its definition is different in detail, so it is necessary to define new properties and addition operations. Formally, the definition of HAS is a tuple of polynomial-time $HAS(Setup, Sign_{sk_i}, Combine, Verify_{cpk})$ [39]. But the signature algorithm is not threshold. Dan et al. [33] introduce a new concept, called a Universal Thresholdizer, from which many threshold systems are possible. We can transform a non-threshold cryptography scheme into a threshold cryptography scheme with the *Universal Thresholdizer* (UT).

A UT scheme is a tuple of algorithms ($UT.Setup$, $UT.Eval$, $UT.Combine$, $UT.Verify$) and needs to satisfy the compactness, correctness, and security properties. Moreover, the verification algorithm of UT needs to satisfy the correctness and robustness. First, we introduce the Universal Thresholdizer of Dan et al. [33].

Definition 5 (Universal Thresholdizer). *Let $P = \{P_1, ..., P_N\}$ be a set of parties. A universal thresholdizer scheme for S and M is a tuple of PPT algorithms* $\mathbf{UT} = (\mathbf{UT}.\mathbf{Setup}, \mathbf{UT}.\mathbf{Eval}, \mathbf{UT}.\mathbf{Combine})$ *with the following:*

- $\mathbf{UT}.\mathbf{Setup}$ $\left(1^\kappa, 1^d, \mathbb{A}, x\right) \rightarrow (\mathrm{pp}, \mathrm{s}_1, \ldots, \mathrm{s}_N)$*: On input the security parameter κ, a depth bound d, an access structure $\mathbb{A}$, and a message $x \in \{0,1\}^k$, the setup algorithm outputs the public parameters* $\mathbf{pp}$*, and a set of shares $s_1, ..., s_N$.*
- $\mathbf{UT}.\mathbf{Eval}$ $(\mathrm{pp}, \mathrm{s}_i, C) \rightarrow \mathrm{y}_i$ *: On input the public parameters* $\mathbf{pp}$*, a share s_i, and a circuit $C : \{0,1\}^k \rightarrow \{0,1\}$ of depth at most d, the evaluation algorithm outputs a partial evaluation y_i.*
- $\mathbf{UT}.\mathbf{Verify}$ $(\mathbf{pp}, y_i, C) \rightarrow \{0,1\}$ *: On input the public parameters* $\mathbf{pp}$*, a partial evaluation y_i, and a circuit $C : \{0,1\}^k \rightarrow \{0,1\}$, the verification algorithm accepts or rejects.*
- $\mathbf{UT}.\mathbf{Combine}$ $(\mathbf{pp}, B) \rightarrow y$ *: On input the public parameters* $\mathbf{pp}$*, a set of partial evaluations $B = \{y_i\}_i \in S$, the combining algorithm outputs the final evaluation y.*

The UT scheme needs to satisfy the compactness, correctness, and security properties. Moreover, the **UT.Verify** need satisfy the correctness and robustness. Specific attribute definitions and proofs can be found in [33] section 7. We next review the lattice-based *Homomorphic Aggregate Signature* (HAS) scheme [39].

Definition 6. *The HAS protocol contains the following four algorithms.*

- **Setup**. *Input public parameter $pp = \{\kappa, N, q, m, n, \sigma, H, h\}$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_q^{Nm}$ be a collision-resistant homomorphic hash function and h is a lattice-based homomorphic hash function. N servers execute the key generation algorithm $\mathcal{F}_{KG}$ to generate public-private key pairs (pk_i, sk_i), where $pk_i, sk_i \in \mathbb{Z}_q^{n\times m}$. Let $PK = \{pk_1, ..., _p k_N\} \in Z_q^{n\times Nm}$.*
- **Sign**. *N servers produce signatures of the message M_i as follows:*
  - *The i-th server calculates $S_i = ExtendBasis(sk_i, PK)$,*
  - *The i-th server calculates $h_i = h(M_i) \bmod q \in \mathbb{Z}_q^n$,*
  - *The i-th server output $Sign_i = SamplePre(PK, S_i, \sigma.h_i) \in Z_q^{Nm}$.*

– **Combine**. *The aggregate signature* $Sign = \sum_{i=1}^{t} a_i \cdot Sign_i$ *is the combined message* $M = \sum_{i=1}^{t} a_i \cdot M_i$*, where* $c_i \in \{0,1\}$*.*
– **Verify**. *The verifier check* $PK \cdot h(Sign) = h(M) \bmod q$ *and* $\|Sign\| \leq N\sigma\sqrt{tm}$*. If both conditions hold, output accept. otherwise, output reject.*

In the correctness and security of the aggregate signature scheme, a set of unique properties need to be considered on the basis of a general linear homomorphic signature. *i*) It is assumed that each verifier is honest. If the signature is valid, the verifier must accept. *ii*) For a series of valid signatures, the aggregate signature can be combined without accessing the server's private key. The HAS scheme needs to satisfy the unforgeable and privacy properties in [39] section 4.

By employing the UT technology and HAS protocol, we can realize a threshold HAS (THAS) protocol as follows:

Definition 7 (THAS). *A THAS protocol contains four algorithms:*

- $(sk_i, pk_i) \leftarrow \mathsf{Setup}(1^\kappa, N)$*. Input the security parameter* 1^κ *and the maximum number of servers* N*.* $\mathcal{F}_{KG}$ *output a private key* sk_i *for each server* S_i*,* $i \in [1, N]$*. Then,* S_i *generates and publish the public key* pk_i*.*
- $Sign_i \leftarrow \mathsf{PartSign}(id, \mathrm{m}_i, sk_i)$*. For the j-th server, input a message* $\mathbf{m}_i$ *of subspace id and* sk_j*. The j-th server outputs the signature* $Sign_i$ *on* $\mathbf{m}_i$*.*
- $Sign \leftarrow \mathsf{Combine}(id, PK, \{\lambda_i, \mathbf{m}_i, Sign_i\}_{i=1}^{t})$*. For* t $(t \leq N)$ *pairs of message sharing the same id and the corresponding signature* $Sign_i$*, output the aggregate signature* $Sign = \sum_{i=1}^{t} \lambda_i Sign_i$ *on message* $\mathbf{m} = \sum_{i=1}^{t} \lambda_i \mathbf{m_i}$ *.*
- $(1, 0) \leftarrow \mathsf{Verify}(id, PK, \mathbf{m}, Sign)$*. Given the id, public key* $PK = \{pk_1 \| ... \| pk_t\}$*, the aggregate message* $\mathbf{m}$*, and the aggregate signature* $Sign$*. If* $PK \cdot Sign = h(\mathbf{m})$ *and* $\|Sign\| \leq n\sigma\sqrt{tm}$*, the client output 1 or 0.*

Remark 1. Notably, the Lagrange coefficients $\lambda_i \in \mathbb{R}$ in the $\mathsf{Combine}$. We can use "clear out their denominators" [33,40] to limit the Lagrange coefficients to integer. For N servers, give t $(t \leq N)$ numbers $I_1, ..., I_t \in [1, N]$. Define the Lagrange coefficients $\lambda_i = \prod_{i \neq j}^{t} \frac{-I_i}{(I_j - I_i)}$. Let the secret space is $\mathbb{Z}_p$ for a series of prime p with $(N!)^3 \leq p$. Then, for every $1 \leq j \leq t$, the integer Lagrange coefficients $\lambda_j = (N!)^2 \cdot \prod_{i \neq j}^{t} \frac{-I_i}{(I_j - I_i)}$ is bounded $\lambda_j \leq (N!)^3$.

Correctness. For any message m_i, $\mathsf{Verify}(id, PK, \mathbf{m}, Sign)$ outputs 1 with overwhelming probability, if S_i strictly implement $(sk_i, pk_i) \leftarrow \mathsf{Setup}(1^\kappa, N)$, $Sign_i \leftarrow \mathsf{PartSign}(id, m_i, sk_i)$, and $Sign \leftarrow \mathsf{Combine}(id, PK, \{\lambda_i, \mathbf{m}_i, Sign_i\}_{i=1}^{t})$.
Security. A THAS is secure if the advantage of any PPT $\mathcal{A}$ without knowing more than t secret keys sk_i to forge a signature Sign^* is that $Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa) \leq \varepsilon(\kappa)$. According to Dan et al. [33], $\mathcal{A}$ cannot get $Sign$ by only executing $\mathsf{Combine}(id, PK, \{\lambda_i, \mathbf{m}_i, Sign_i\}_{i=1}^{t})$. For the aggregate signature Sign on M, $\mathcal{A}$ can outputs a list of query messages $M = \{m_1, ..., m_Q\}$, then query $Sign_i \leftarrow \mathsf{PartSign}(id, m_i, sk_i)$ to obtain at least a $\mathsf{Sign_i}$. By [39,41], the HAS meets unforgeability and binding. It means that $\mathcal{A}$ cannot get the $Sign_i$ of S_i for m_i unless $\mathcal{A}$ knows sk_i.

For a security parameter κ and public parameter pp, the functionality $\mathcal{F}_{TOPRF}$ interacts with a set of servers $S = \{S_1, ..., S_n\}$, an arbitrary client C, and an adversary $\mathcal{A}$. We define $F_k(\mathbf{x})$ as a fix PRF function.Initialize counters ψ to 0.

Key Generation

On receiving $\mathcal{F}_{KG}$ from S_i:

– If $\mathcal{F}_{KG}$ was received from all S_i, give output key share k_i to all S_i.

Evaluation

On receiving $(Blind, ID_c, value_c)$ from any client C

– Record $(eval, ID_c, C, pp, value_c, \perp)$, and send output $(Setup, ID_c, pp)$ to every S_i.

On receiving $(Derive, ID_c)$ from S_i

– Retrieve record $(eval, ID_c, C, pp, value_{s_i})$.
– If $(Derive, ID_c)$ has been received from all S_i, set ψ to $\psi + 1$.

On receiving $(Query, ID_c)$ from $\mathcal{A}$

– Retrieve record $(eval, ID_c, C, pp, value_{s_i})$, only proceed if $\psi > 0$, set ψ to $\psi - 1$.
– If $value_c$ is undefined, then pick $\rho \overset{R}{\leftarrow} \{0,1\}^{\kappa}$, and set $value_c$ to ρ.
– Output $(Query, ID_c, pp, value_c)$ to C.

Fig. 1: The Ideal Functionality $\mathcal{F}_{\mathsf{TOPRF}}$.

2.4 Oblivious Pseudorandom Function over Lattices

Let $\ell = \lceil \log_2 q \rceil$. Define $G : R_q^{\ell\times\ell} \to R_q^{1\times\ell}$ to be the linear operation corresponding to left multiplication by $(1, 2, ..., 2^{\ell-1})$ and the inverts $G^{-1} : R_q^{1\times\ell} \to R_q^{\ell\times\ell}$. Fix an array of $\mathbf{a}_0, \mathbf{a}_1 \overset{R}{\leftarrow} R_q^{1\times\ell}$. $G^{-1}(\mathbf{a})$ can be regarded as the bit decomposition operation of $\mathbf{a}$ into binary polynomials. For any $\mathbf{x} = (x_1, ..., x_L) \in \{0,1\}^L$ subject to $\mathbf{a}_x := \boldsymbol{a}_{x_1} \cdot G^{-1}\left(\boldsymbol{a}_{x_2} \cdot G^{-1}\left(\ldots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\right)\right) \in R_q^{1\times\ell}$.

Lemma 1 (PRF, [42]). *Sample $k \overset{\$}{\leftarrow} \mathbb{Z}_q$, the function $F_k(\mathbf{x}) = \lfloor \frac{p}{q} \cdot \mathbf{a}_x \cdot k \rceil$ is a PRF over the $DRLWE_{q,n,\sigma}$ if $q \gg p \cdot \sigma \cdot n \cdot \ell \cdot \sqrt{L}$.*

Lemma 2 (Bound on errors, [34]). *Let $x \in \{0,1\}^L, \ell = \lfloor \log_2 q \rceil$ and $n = poly(\kappa)$. Sample from the probability distribution space of error ε_σ have infinity norm at most $L \cdot \ell \cdot \sigma \cdot n^{3/2}$ with all but negligible probability.*

PTA schemes are used to hide the password by TOPRF. Multi-servers assist the client in computing the PRF value on the input, but learning nothing about the client's input to prevent offline password guessing attacks. The main goal of our work in this section is to build a TOPRF, and declare a provably secure structure in the MPC model. We define an ideal functionality $\mathcal{F}_{\mathsf{TOPRF}}$ in Fig. 1.

Each server executes $\mathcal{F}_{KG}$ (Definition 4) to obtain a secret sk_i, and assist the client entering $\mathbf{x}$ to generate $F_k(x)$. It is challenging to construct a programmable random oracle in the QROM, known to be difficult[43], so we employ the MPC model [34] instead of the UC model [17] to define the security of TOPRF.

Definition 8 ([34]). *Let $\mathcal{K}$ denote the key distribution. The set of servers $\mathbb{S}$ and clients $\mathbb{C}$ are two parties of TOPRF protocol Π. For $k \overset{\$}{\leftarrow} \mathcal{K}$, there is $\mathsf{real}_{\Pi,\mathcal{A},\mathbb{C}}(\mathbf{x}, \mathcal{K}, 1^\kappa)$ to denote the joint output distribution of $\mathcal{A}(\mathbf{x})$ and $\mathbb{S}(k)$. Sim is a PPT simulator. A protocol Π is a TOPRF if the following holds:*

- **Correctness** : *For every inputs $(\mathbf{x}, k_i)$, $\mathsf{Pr}[\ \Pi\ (\mathbf{x}, k) \neq F_k(\mathbf{x})] \leq \varepsilon(\kappa)$.*
- **Malicious client security(obliviousness)** : *For any PPT $\mathcal{A}$ corrupting $t' \leq t$ server, there exists a PPT simulator Sim such that for every pair of inputs $(\mathbf{x}, k)$: $\mathsf{ideal}_{\mathcal{F}_{\mathrm{TOPRF}},Sim,\mathcal{A},\mathbb{S}}(\mathbf{x}, k, 1^\kappa) \approx_c \mathsf{real}_{\Pi,\mathcal{A},\mathbb{S}}(\mathbf{x}, k, 1^\kappa)$.*
- **Average case malicious server security(unpredictability)** : *For any PPT $\mathcal{A}$ corrupting a client, there exists a PPT simulator Sim such that for all clients input x, $\mathsf{ideal}_{\mathcal{F}_{\mathrm{TOPRF}},Sim,\mathcal{A},\mathbb{C}}(\mathbf{x}, \mathcal{K}, 1^\kappa) \approx_c \mathsf{real}_{\Pi,\mathcal{A},\mathbb{C}}(\mathbf{x}, \mathcal{K}, 1^\kappa)$ And if $\mathcal{A}$ correctly outputs $F_k(\mathbf{x})$ with all but negligible probability over the choice $k \leftarrow \mathcal{K}$ when interacting directly with $\mathbb{S}(k)$ using Π, then $\mathcal{A}$ also outputs $F_k(\mathbf{x})$ with all but negligible probability when interacting via Sim.*

Oblivious Computation of PRF $F_k(\mathbf{x})$ between client C and server S

1. On input $\mathbf{x}$, C chooses $s \xleftarrow{R} Z_q^{n\times 1}$ and $e \xleftarrow{R} \mathcal{X}^{m\times 1}$; sends $\mathbf{x}^* = A\cdot s + e + \mathbf{a}^F(\mathbf{x})$ to each S_i.
2. S_i chooses $e_i' \xleftarrow{R} \mathcal{X}^{m\times 1}$ responds with $\mathbf{x}^*_{k_i} = \mathbf{x}^* \cdot k_i + e_i'$.
3. C computes $PK = \sum_{i=1}^{t} \lambda_{i,j} \cdot pk_i$, and output

$F_{msk}(\mathbf{x}) = \lfloor \frac{p}{q}(\sum_{i=1}^{t} \lambda_{i,j} \cdot \mathbf{x}^*_{sk_i} - PK \cdot s) \rceil \approx \lfloor \frac{p}{q} \cdot \mathbf{a}^F(\mathbf{x}) \cdot msk \rceil$.

Fig. 2: The TOPRF Algorithm $\mathit{\Pi}_{\mathsf{TOPRF}}$.

For a particular function $\mathbf{a}^F : \{0,1\}^L \to R_q^{1\times L}$, we employ a bottom PRF is essentially the a particular instantiation of the PRF [42] $F_k(\mathbf{x}) = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(\mathbf{x}) \cdot k \right\rceil$. All servers execute $\mathcal{F}_{KG}$, which is introduced in the definition 4, to generate the secret key k_i for S_i and these secret key k_i correspond to a master private key msk. For a public matrix $A \in R_q^{m\times n}$, each server S_i publishes their public key $pk_i := A \cdot k_i + e_i$, where $e_i \xleftarrow{R} \mathcal{X}^{m\times 1}$. For $e, e_i, e_i' \le \sigma\sqrt{n}$ and $e'' = msk \cdot e + \sum_{i=1}^{n} \lambda_{i,j} e_i' - \sum_{i=1}^{n} e_i \le L \cdot \ell \cdot \sigma \cdot n^{3/2}$, a details of $\mathit{\Pi}_{\mathsf{TOPRF}}$ is in Fig. 2.

Lemma 3. *Let $q, m, n, \sigma > 0$ depend on κ and $\ell = \lceil \log_2 q \rceil$. For any noise $e \le \sigma\sqrt{n}$, the $\mathbf{x}$ entered by the client is converted into binary and write as $\mathbf{x} = (x_1, ..., x_L) \in \{0,1\}^L$. Our TOPRF can obtain indistinguishable outputs for the same input with a reasonable choice of parameters, if $N \le \frac{1}{4} \log_2 \frac{L\cdot\ell\cdot n - \sigma\sqrt{n}}{\sigma\sqrt{n} - 1}$.*

Proof. C chooses $s \xleftarrow{R} \mathbb{Z}_q^{n\times 1}$ and $e \xleftarrow{R} \mathcal{X}^{m\times 1}$. Each S_i chooses $e_i, e_i' \xleftarrow{R} \mathcal{X}^{m\times 1}$. Both sides of the interaction execute $\mathit{\Pi}_{\mathsf{TOPRF}}$ and C output $F_{msk}(\mathbf{x})$. Let $e'' = msk \cdot e + \sum_{i=1}^{t} \lambda_i e_i' - \sum_{i=1}^{t} e_i$, Let λ_i denote the Lagrange coefficient such that $msk = \sum_{i=1}^{t} \lambda_i k_i$, we have: $F_{msk}(\mathbf{x}) = \lfloor \frac{p}{q}(\sum_{i=1}^{t} \lambda_i \cdot x^*_{k_i} - PK \cdot s) \rceil = \lfloor \frac{p}{q}(((A\cdot s + e + \mathbf{a}^F(\mathbf{x})) \cdot \sum_{i=1}^{t} \lambda_i k_i + \sum_{i=1}^{t} \lambda_i e_i') - \sum_{i=1}^{t}(A \cdot k_i + e_i) \cdot s) \rceil = \lfloor \frac{p}{q}(\mathbf{a}^F(\mathbf{x}) \cdot msk + e'') \rceil$.

By Lemma 1, we know that the function $\lfloor \frac{p}{q} \cdot \mathbf{a}_x \cdot k \rceil$ is a PRF over the DRLWE$_{q,n,\sigma}$ when $q \gg p \cdot \sigma \cdot n \cdot \ell \cdot \sqrt{L}$. That means $\lfloor \frac{p}{q} \cdot \mathbf{a}_x \cdot k \rceil \approx_c \lfloor \frac{p}{q} u \rceil$, where u is uniform in $R_q^{1\times \ell}$. From Lemma 2, let $e'' \le L \cdot \ell \cdot \sigma \cdot n^{3/2}$, we have $F_{msk}(\mathbf{x}) \approx \lfloor \frac{p}{q} \mathbf{a}^F(\mathbf{x}) \cdot msk \rceil \approx_c \lfloor \frac{p}{q} u \rceil$. Next, we analyze e''. We take the maximum value of each term in e'' and $|e''| = |\sigma^2 \cdot n + N(N!)^3 \cdot \sigma\sqrt{n} - N(N!)^3 \cdot \sigma^2 \cdot n| \le L \cdot \ell \cdot \sigma \cdot n^{3/2}$. The fact that $(N!)^3 \le (N)^{3N} \le 2^{4N}$, thus $|e''| \le |\sigma^2 \cdot n + 2^{4N+1}(\sigma\sqrt{n} - \sigma^2 \cdot n)| \le L \cdot \ell \cdot \sigma \cdot n^{3/2}$. We have $N \le \frac{1}{4} \log_2 \frac{L\cdot\ell\cdot n - \sigma\sqrt{n}}{\sigma\sqrt{-1}}$. To sum up, let $q \gg p \cdot \sigma \cdot n \cdot \ell \cdot \sqrt{L}$ and $N \le \frac{1}{4} \log_2 \frac{L\cdot\ell\cdot n - \sigma\sqrt{n}}{\sigma\sqrt{-1}}$, the PRF $F_{msk}(\mathbf{x}) = \lfloor \frac{p}{q} \mathbf{a}^F(\mathbf{x}) \cdot msk \rceil$. □

Security. Our TOPRF inherits the security properties of the definition 8 and satisfies quantum security, unpredictability, and obliviousness. Quantum security ensures that no adversary with any quantum capability can access the hidden secrets by cracking the encryption algorithm. Unpredictability and obliviousness mean that the value of the PRF is independent of the input secret (unpredictability and obliviousness correspond to average case malicious server security and malicious client security in definition 8, respectively), which makes TOPRF resistant to offline password guessing attacks. We prove the stored secret information sd_i only depends on server-side secrets sk_i by introducing unpredictability and obliviousness.

Unpredictability. $F_{msk}(\mathbf{x})$ is unpredictability, which means that the adversary $\mathcal{A}$ uses $\mathbf{x}$ to interacte with a set of servers and cannot predict the value of

$F_{msk}(\mathbf{x})$, even if the adversary $\mathcal{A}$ can corrupt $t' < t$ servers. Unpredictability corresponds to average-case malicious client security in definition 8. We describe a simulation Sim that communicates with malicious client C^* and $\mathcal{F}_{TOPRF}$. Specifically, Sim carries out the following steps:

– In the initialization phase, $\mathcal{A}$ and uniform $pk_{\mathcal{A}} \overset{R}{\leftarrow} R_q^{1\times\ell}$ are generated. Send public parameters pp and pk_i to $\mathcal{A}$. Initialise an empty list $\mathbf{Q}$.
– During the query stage, for each message pk_i, do: $\mathcal{A}$ extracts $\mathbf{x}_{\mathcal{A}}, e_{\mathcal{A}}$, and sends the queries $\mathbf{x}$ to the functionality $\mathcal{F}_{TOPRF}$. If $\mathcal{F}_{TOPRF}$ returns $F_{msk}(\mathbf{x}) \in R_p^{1\times\ell}$ and $F_{msk}(\mathbf{x}) \notin \mathbf{Q}$, sample $\boldsymbol{F}_q \overset{\$}{\leftarrow} R_q^{1\times\ell} \cap \left(\frac{q}{p}\boldsymbol{y} + R^{1\times\ell}_{\leq\frac{q}{2p}}\right)$ and add $(\mathbf{x}, \boldsymbol{F}_q)$ into $\mathbf{Q}$. Return $\boldsymbol{F}_q$ to $\mathcal{A}$. If $\mathcal{F}_{TOPRF}$ returns $F_{msk}(\mathbf{x}) \in R_p^{1\times\ell}$ and $F_{msk}(\mathbf{x}) \in \mathbf{Q}$, set $\boldsymbol{F}_q = F_{msk}(\mathbf{x}) \in R_p^{1\times\ell}$. Choose $e_i^* \overset{R}{\leftarrow} \chi_{\sigma'}$ and send $\mathbf{x}^*_{k_i} = pk \cdot k_i + e_i^* + \boldsymbol{F}_q$ to $\mathcal{A}$. Each round of queries uses different errors sampled from $R(\chi_{\sigma'}^{1\times\ell})$. In a real protocol, if the adversary $\mathcal{A}$ can calculate the correct $\boldsymbol{F}_q$, it can perform the same operation on the message received from the simulator. $\boldsymbol{F}_q$ is sampled by Sim and the corresponding value $\mathbf{x}^*_{k_i}$. Let $e_{\lfloor\rceil} := \boldsymbol{y}_q - (q/p)\cdot\boldsymbol{y} \in R^{1\times\ell}_{\leq\frac{q}{2p}}$, we have $\lfloor\frac{p}{q}(\sum_{i=1}^{N}\lambda_i\cdot\mathbf{x}^*_{k_i} - PK\cdot s)\rceil = \lfloor\frac{p}{q}(\mathbf{a}^F(\mathbf{x})\cdot msk + e_{\lfloor\rceil} + e'')\rceil$, where $e'' \leq L\cdot\ell\cdot\sigma\cdot n^{3/2}$. Let $T = L\cdot\ell\cdot\sigma\cdot n^{3/2}$, there is $\|e_{\lfloor\rceil}\| < q/(2p) - T$. Thus, $\|msk\cdot e + \sum_{i=1}^{N}\lambda_i e_i' - \sum_{i=1}^{N} e_i\| \leq \frac{1}{2}$.

Obliviousness. $F_{msk}(\mathbf{x})$ is obliviousness, which means that even if $\mathcal{A}$ gets the value of $F_{msk}(\mathbf{x})$, it also cannot learn anything about the input $\mathbf{x}$, even if $\mathcal{A}$ knows k_i. Obliviousness corresponds to average-case malicious server security in definition 8. We describe a simulation Sim that communicates with $\mathcal{A}$ and the functionality $\mathcal{F}_{TOPRF}$. In the initialization phase, $\mathcal{A}$ plays malicious server S^*. S^* computes pk^* from k^* and publishes it, where $k_i \leq \sigma\cdot\sqrt{n}$. In the query phase, Sim randomly selected $r \overset{R}{\leftarrow} R_q^{1\times\ell}$ and send to S^* . Waiting for a response of $\mathbf{x}^*_{k_i}$ from S^*. Finally, the honest client C send $F_{msk}(\mathbf{x})$ to adversary $\mathcal{A}$.

In real protocol, $\mathbf{x}^*$ generated by the honest client C. The secret value $\mathbf{x}$ is hidden by the encryption algorithm based on DRLWE. Thus, $\mathcal{A}$ cannot distinguish a real $\mathbf{x}^*$ from r. Let $\mathbf{x} \overset{R}{\leftarrow} R(\chi_\sigma)$ and $e \overset{R}{\leftarrow} R(\chi_\sigma)^{1\times\ell}$ are sampled by C. For C interacting with $\mathcal{F}_{TOPRF}$, and computes $F_{msk}(\mathbf{x}) = \lfloor\frac{p}{q}(\sum_{i=1}^{N}\lambda_i\cdot\mathbf{x}^*_{k_i} - PK\cdot s)\rceil = \lfloor\frac{p}{q}(\mathbf{a}^F(\mathbf{x})\cdot msk + e'')\rceil$, where $e'' = msk\cdot e + \sum_{i=1}^{N}\lambda_i e_i' - \sum_{i=1}^{N} e_i$. If $N \leq \frac{1}{4}log_2\frac{L\cdot\ell\cdot n - \sigma\sqrt{n}}{\sigma\sqrt{n}-1}$, the coefficient of $\frac{p}{q}\cdot\mathbf{a}^F(\mathbf{x})\cdot msk$ is further than e'' away from $\mathbb{Z}+\frac{1}{2}$. According to $Adv_{\mathcal{A}}^{\mathsf{DRLWE}}(\kappa) \leq \varepsilon(\kappa)$, $\mathcal{A}$ can learn nothing about $\mathbf{x}$.

3 Basic Scheme Architecture and Security Model

3.1 Password-based Threshold SSO Authentication

We propose a password-based threshold SSO authentication architecture, as shown in Fig. 3, including the following two entities.

– **Client**. The client (denoted by C) registers with identity servers using the ID_c and a human-memorable password psw_c. When the client enters the correct password psw_c login corresponding to the ID_c, an authentication token can be obtained from identity servers.

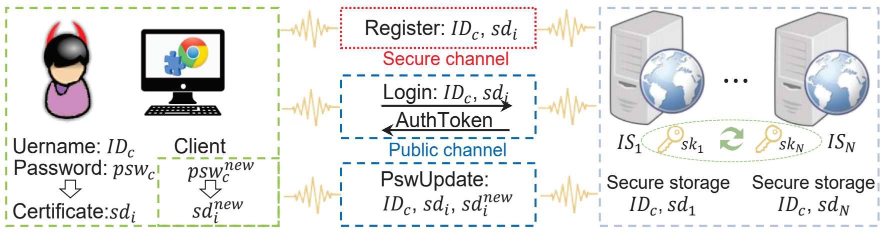


Fig. 3: The Basic Scheme Architecture.

– **Identity servers**. There is a set of identity servers (denoted by $IS = \{IS_1, ..., IS_N\}$) that provide authentication tokens to the client. Each server stores a portion of the registration information independent of the client's password and generates an authentication token for the authenticated client.

Our scheme mainly focuses on the interaction between the client and a series of identity servers, including the interaction in the registration phase and login authentication phase. In addition, we add two dynamic update functions: password and server private key update, which further improves the security of the scheme. The formal definition of the scheme's functionality is as follows.

Definition 9. *The quantum-resistant password-based threshold SSO authentication scheme over lattices includes the following five polynomial algorithms:*

– $pp \leftarrow \mathsf{Setup}(1^\kappa)$. *The algorithm generates the set of system parameters pp by input a security parameter κ, including hash algorithm, common matrix, discrete Gaussian sampling parameters, etc., for C and IS. $IS_i(i \in [1,n])$ calculates the public-private key pair (sk_i, pk_i) and publishes pk_i. (The sk_i is generated by F_{KG} in Definition 4).*

– Register *is a registration protocol executed between the client C and a set of servers IS according to the following specification:*
$\{0,1\} \leftarrow \mathsf{RegC}(pp, ID_c, psw_c, IS_i)$. *C interacts with multiple servers to register by psw_c and ID_c. If the registration aborts, the algorithm outputs 0. Otherwise it output 1.*
$\{0,1\} \leftarrow \mathsf{RegS}(pp, ID_c, sk_i)$. *$IS_i(i \in [1,N])$ helps C to get a server-derived key sd_i and hold it. If the registration fails, the algorithm outputs 0. Otherwise it outputs 1 and stores the user's identity information.*

– Login *is a login and authentication protocol executed between the client C and a set of servers IS according to the following specification:*
$AutToken \leftarrow \mathsf{LoginC}(pp, ID_c, psw_c, IS_i)$. *C logins with psw_c and ID_c from the registration phase. Then, C verifies Aut_i from IS_i and aggregates t valid Aut_i to generate an authentication token AutToken.*
$Aut_i \leftarrow \mathsf{LoginS}(pp, ID_c, sk_i)$. *$IS_i$ assists C to generate sd'_i and compare it with sd_i stored in the registration phase. If $sd'_i \neq sd_i$, login aborts. Otherwise, IS_i generates $Aut_i = \mathsf{THAS}.PartSgin(ID_c, H(Token), sk_i)$ for C, where Token represents the client's message with client attributes, service validity time, access control policy, and other auxiliary information.*

– PswUpdate *is a password update protocol executed between the client C and a set of servers IS according to the following specification:*
$\{0,1\} \leftarrow \mathsf{PswUpdate}(pp, ID_c, psw_c^{old}, psw_c new, IS_i)$. *C interacts with multiple servers to modify the password. C generates sd_i^{old} and sd_i^{new} for psw_c^{old}*

and psw_c^{new}, respectively, by interacting with IS_i and sending them to IS_i. If password update fails, the algorithm outputs 0. Otherwise it output 1.
$\{0,1\} \leftarrow \mathsf{PswUpdate}(pp, ID_c, sk_i)$. *$IS_i$ assists C to generate sd_i^{old} and sd_i^{new}. IS_i compares sd_i^{old} with sd_i stored in the registration phase. If $sd_i^{old} \neq sd_i$, login aborts and output 0. Otherwise, IS_i stores sd_i^{new} and return 1.*

– $(0,1) \leftarrow \mathsf{SkUpdate}(sk_i, Q)$. *Each IS_i periodically updates sk_i to resist permanent corruption attacks. IS can update the private key by the update polynomial Q without client participation. The generated authentication tokens are not invalidated after the complete update.*

Correctness. The correctness of the scheme above means that C can obtain vaild authentication token $Token$ if C inputs the psw_c consistent with the registration phase in the login phase. Formally, for any honest C, the probability $Pr[AutToken \leftarrow \mathsf{LoginC}(pp, ID_c, psw_c, IS_i)] = 1$ iff $\{0,1\} \leftarrow \mathsf{RegC}(pp, ID_c, psw_c, IS_i)$, $\{0,1\} \leftarrow \mathsf{RegS}(pp, ID_c, sk_i)$, $Aut_i \leftarrow \mathsf{LoginS}(pp, ID_c, sk_i)$.

3.2 Security Model

We consider an adversary with quantum computing power and controlling $t' \leq t$ servers in an epoch[5]. The adversary executes both online and offline guessing attacks to capture the authentication information of the honest client. We assume that the adversary needs to be profitable to launch an attack, as in [12,18,20].

Concretely, our single sign-on system assumes that the adversary's goal is to obtain an authentication token. There are two attack ways for the adversary, i.e., i) guessing the user's password and obtaining the authentication token by the login; ii) obtaining a sufficient number (greater than the threshold t) of server-side keys and thus forging the authentication token. This implies that a malicious server will not intentionally execute the protocol incorrectly to cause the user to generate the wrong authentication token, since this is not profitable for the adversary. But it is more desirable if the computational legitimacy of the scheme can be verified. Zero-knowledge proofs may be a available idea, but they are too expensive for single sign-on. We leave this as our future work. We employ the *Bellare-Pointcheval-Rogaway* (BPR)-like model [35] to analyze the security of our password-based scheme. Let L denote a maintained list by the experiments. sid_i denote i-th session. We define the following experiment:

```
$\boldsymbol{Experiment}\ \ Exp^{Auth}_{PTA,\mathcal{A}}(\kappa)$ :
$sd_i \leftarrow \emptyset; sid_j \leftarrow 0; pp \leftarrow Setup(1^\kappa);$
$(sid_{i^*}, ID^*, psw^*) \leftarrow \mathcal{A}^{oracle(\cdot)}(pp);$
$Aut_i \leftarrow \mathsf{LoginS}(sid_{i^*}, ID^*, psw^*);$
$(sid_{i^*}, ID^*, psw^*) \notin L \wedge (psw^* \in |D|)$ return 1
else rutrun 0
```

where the experiment uses the following oracles:

- $\mathsf{Challenge}(c, sid_i, ID_c)$: The oracle aborts if $(sid_{i^*} \geq 0) \vee (sid_i \geq sid_j) \vee ((sid_i, ID_c) \in L)$. Otherwise, it set $sid_{i^*} \leftarrow sid_i$ and access oracle LoginC.

5 To cope with perpetual leakage [44], each server renews its secret key in a fixed time interval, called an epoch.

- Registration(i) The experiment randomly picks psw satisfy $(sid_i, psw, i) \notin L$. At this point, $\mathcal{A}$ interacts with the honest client and server (oracle) as the corrupted server. After access, the experiment records $L[sid_i] \leftarrow (i, psw, sd_i)$, delivers j to $\mathcal{A}$ and set $j \leftarrow j+1$.
- RegistrationS(sid_i): The oracle aborts if $sid_i \geq sid_j$. Otherwise, it gets $(i, psw, sd_i) \leftarrow L[sid_i]$. $\mathcal{A}$ interacts with the honest server (oracle) as the corrupted server.
- LoginC(sid_i, ID, psw): The oracle aborts if $sid_i \geq sid_j$. Otherwise, it gets $(i, psw, sd_i) \leftarrow L[sid_i]$. $\mathcal{A}$ interacts with the honest client and server (oracle) as the corrupted server. In authentication experiment, the oracle additionally computes $L \leftarrow L \cup (sid_i, ID, psw)$.
- LoginS(sid_i): The oracle aborts if $sid_i \geq sid_j$. Else, it gets $(i, psw, sd_i) \leftarrow L[sid_i]$. $\mathcal{A}$ interacts with the honest server (oracle) as the corrupted server.

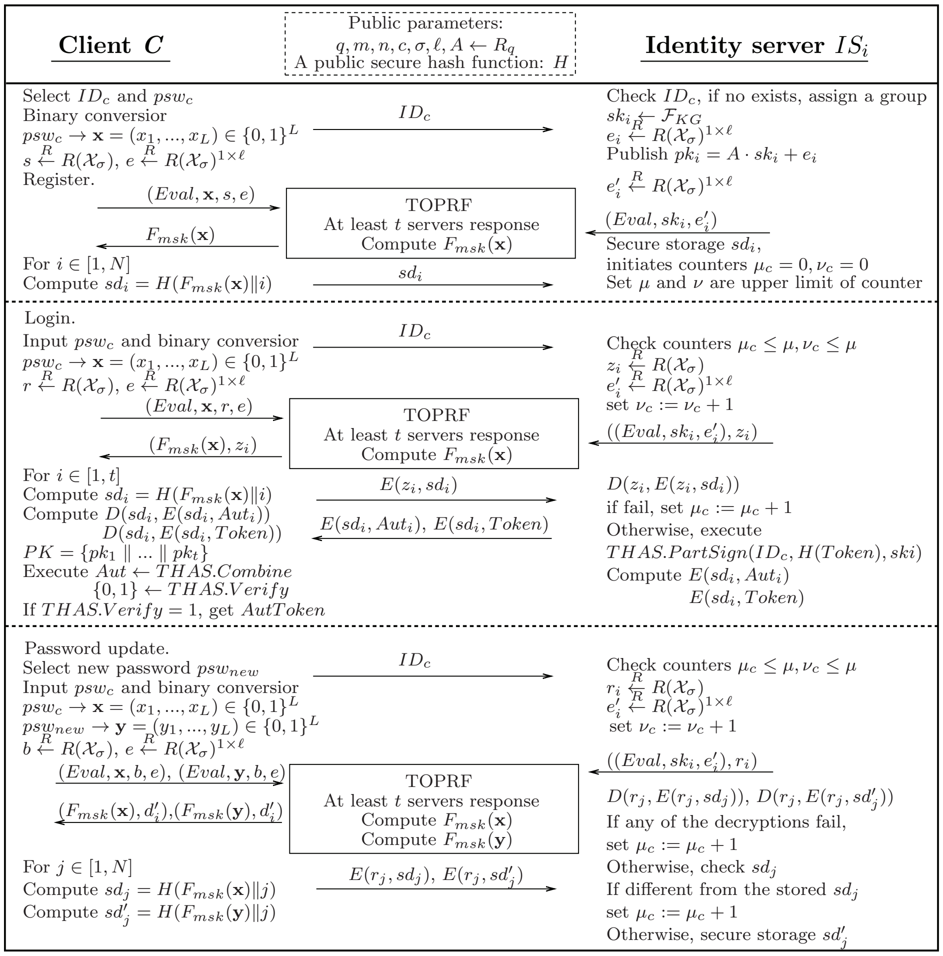
The password update can be considered as combining a login algorithm without issuing a token and a registration algorithm. Consequently, we mainly focus on the registration and login phase. Notably, it has been shown that passwords follow the CDF-Zipf distribution [45]. We recommend using the accurate Zipf-based formulation for our password-based scheme.

Definition 10. *Assuming that a PPT $\mathcal{A}$ executes at most q_{send} online attacks, the advantage of $\mathcal{A}$ denoted $C' \cdot q_{send}^{s'}(\kappa) + \varepsilon(\kappa)$ where C' and s' are the Zipf parameters for all dictionary sizes $|\mathcal{D}|$ in the Zipf-law [45]. Following the experiment $Exp_{PTA,\mathcal{A}}^{Auth}(\kappa)$, the advantage of our scheme holds that $Adv_{PTA,\mathcal{A}}^{Auth}(\kappa) = Pr[1 \leftarrow Exp_{PTA,\mathcal{A}}^{Auth}(\kappa)] \leq C' \cdot q_{send}^{s'}(\kappa) + \varepsilon(\kappa)$.*

Definition 11. *To the best of our knowledge, the Grover algorithm, which is currently the most efficient for symmetric cryptosystems under the quantum computing model, only reduces the effective length of the key to half of the original one. Hence AES-256 still has 128-bit security in quantum attacks. For any PPT $\mathcal{A}$, we define the advantage $Adv_{\mathcal{A}}^{\mathsf{AES}}(\kappa) \leq \varepsilon(\kappa)$.*

4 Quantum-Resistant Password-Based Threshold Single-Sign-On Authentication with Secret Update

The quantum-resistant password-based threshold SSO authentication scheme focus on the interaction between a client C, and a set of servers $IS = \{IS_1, ..., IS_N\}$. The details of the interaction part in our protocol are shown in Fig. 4. We highlight how clients can register, login, and password update by interacting with the server based on the TOPRF and THAS over lattices. In the Setup, with the security parameter κ, the system generates public parameters $A \in R_q^{1\times\ell}$, $\sigma > 0$, and $c \in R_q$. Set $m > c\kappa log_2(q)$ and $q \geq poly(\kappa)(\sqrt{log_2\kappa})$. H is a family of collision resistant hash functions that preserve homomorphism property for verification [46]. Let μ be an upper limit that a client fails to pass IS_i authentication, and an upper limit ν is the number of authentication token requests issued by a client in an epoch. E is a secure symmetric key encryption algorithm



Client C
Public parameters:
$q, m, n, c, \sigma, \ell, A \leftarrow R_q$
A public secure hash function: H
Identity server IS_i
Select ID_c and psw_c
Binary conversion
$psw_c \rightarrow \mathbf{x} = (x_1, ..., x_L) \in \{0,1\}^L$
$s \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)$, $e \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
Register.
ID_c
Check ID_c, if no exists, assign a group
$sk_i \overset{R}{\leftarrow} \mathcal{F}_{KG}$
$e_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
Publish $pk_i = A \cdot sk_i + e_i$
$e'_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
$(Eval, \mathbf{x}, s, e)$
TOPRF
At least t servers response
Compute $F_{msk}(\mathbf{x})$
$(Eval, sk_i, e'_i)$
$F_{msk}(\mathbf{x})$
Secure storage sd_i,
initiates counters $\mu_c = 0, \nu_c = 0$
Set μ and ν are upper limit of counter
For $i \in [1, N]$
Compute $sd_i = H(F_{msk}(\mathbf{x})\|i)$
sd_i
Login.
Input psw_c and binary conversion
$psw_c \rightarrow \mathbf{x} = (x_1, ..., x_L) \in \{0,1\}^L$
$r \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)$, $e \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
ID_c
Check counters $\mu_c \leq \mu, \nu_c \leq \mu$
$z_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)$
$e'_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
set $\nu_c := \nu_c + 1$
$(Eval, \mathbf{x}, r, e)$
TOPRF
At least t servers response
Compute $F_{msk}(\mathbf{x})$
$((Eval, sk_i, e'_i), z_i)$
$(F_{msk}(\mathbf{x}), z_i)$
For $i \in [1, t]$
Compute $sd_i = H(F_{msk}(\mathbf{x})\|i)$
Compute $D(sd_i, E(sd_i, Aut_i))$
$D(sd_i, E(sd_i, Token))$
$E(z_i, sd_i)$
$D(z_i, E(z_i, sd_i))$
if fail, set $\mu_c := \mu_c + 1$
Otherwise, execute
$THAS.PartSign(ID_c, H(Token), ski)$
Compute $E(sd_i, Aut_i)$
$E(sd_i, Token)$
$E(sd_i, Aut_i)$, $E(sd_i, Token)$
$PK = \{pk_1 \| ... \| pk_t\}$
Execute $Aut \leftarrow THAS.Combine$
$\{0,1\} \leftarrow THAS.Verify$
If $THAS.Verify = 1$, get $AutToken$
Password update.
Select new password psw_{new}
Input psw_c and binary conversion
$psw_c \rightarrow \mathbf{x} = (x_1, ..., x_L) \in \{0,1\}^L$
$psw_{new} \rightarrow \mathbf{y} = (y_1, ..., y_L) \in \{0,1\}^L$
$b \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)$, $e \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
ID_c
Check counters $\mu_c \leq \mu, \nu_c \leq \mu$
$r_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)$
$e'_i \overset{R}{\leftarrow} R(\mathcal{X}_\sigma)^{1\times\ell}$
set $\nu_c := \nu_c + 1$
$(Eval, \mathbf{x}, b, e)$, $(Eval, \mathbf{y}, b, e)$
TOPRF
At least t servers response
Compute $F_{msk}(\mathbf{x})$
Compute $F_{msk}(\mathbf{y})$
$((Eval, sk_i, e'_i), r_i)$
$(F_{msk}(\mathbf{x}), d'_i)$,$(F_{msk}(\mathbf{y}), d'_i)$
$D(r_j, E(r_j, sd_j))$, $D(r_j, E(r_j, sd'_j))$
If any of the decryptions fail,
set $\mu_c := \mu_c + 1$
Otherwise, check sd_j
If different from the stored sd_j
set $\mu_c := \mu_c + 1$
Otherwise, secure storage sd'_j
For $j \in [1, N]$
Compute $sd_j = H(F_{msk}(\mathbf{x})\|j)$
Compute $sd'_j = H(F_{msk}(\mathbf{y})\|j)$
$E(r_j, sd_j)$, $E(r_j, sd'_j)$

Fig. 4: The registration, login, and update phases of our scheme.

(eg., AES-256), and D denote corresponding decryption algorithm. We write N as the total number of servers, and t is the threshold number.

Notice that in the threshold oblivious pseudorandom function, if the final calculation of the client is a PRF value, it is necessary to limit the size of the coefficient of noise e. The only variable in the noise factor is the number of thresholds (more details of the derivation of the conclusions and the limits of the noise in Lemma 3). Hence, the noise can be effectively reduced by grouping users and using a server-side master key for each group of users. In addition, grouping users and limiting the number of logins within a group in an epoch (without grouping, it is unrealistic to restrict the number of logins for all users [12]) can effectively weaken the impact of online dictionary attacks and reduce the computational cost of the client, and the storage cost of the server [18].

Concretely, in the registration phase, after receiving the ID_c, the server determines whether the ID is a duplicate. If not, it stores the ID and assigns it a group. If no group can receive the client, a new group is generated for it. And settle the new server private key through the F_{KG}. In Login, if THAS.Verify output 1, C uses $AutToken = \{Aut, Token, I\}$ as an authentication token, where I is the set of signature servers' identifiers. The service provider can check the validity of $AutToken$ by executing $(1, 0) \leftarrow \mathsf{Verify}(id, PK, Token, Aut)$ in Definition 7.

To cope with perpetual leakage [44], each IS_i should update sk_i at the end of an epoch. We describe SkUpdate at the end of the ω-th epoch as follows.

- US1. Let $q = e(S)$, IS_j randomly chooses a polynomial $[U] = \sum_{k=1}^{t-1} \alpha_k X^k$ over $R = \mathbb{Z}_q[X]/F(X)$, where $[U]^0 = 0$.
- US2. At least t servers IS_j computes $\{H_j(\alpha_k)\}_{k\in[1,t-1]}$ and $U_j^i = [U]_j^i \ mod \ q$, where $i \in [1, N], j \in [1, t]$.
- US3. IS_j broadcast the message $U_v^{(\omega)} = \{j, \omega, \{H_j(\alpha_k)\}_{k\in[1,t-1]}, E(i, U_j^i)\}$ and the signature $Sign_j \leftarrow PartSign(id, U_v^{(\omega)}, sk_j)$.
- US4. IS_i decrypts the shares intended $\{U_j^i\}_{j\in[1,t]}$ for IS_i and verifies the correctness of the share by checking the equivalent $H(U_j^i) = \sum_{k=1}^{t-1} H(\alpha_k)^{i^k}$ and executing $\mathsf{Verify}(id, pk_i, [U]_j^i, Sign_j)$ (It can be seen as Combine phase signature and message aggregation with 0). If Verify output 1 and the equation holds, each IS_i computes a new $sk_i' = sk_i + \sum_{j=1}^{t} \lambda_{i,j}[U]_j^i \ mod \ q$.
- US5. IS_i recalculates $pk' = A \cdot sk_i' + e_i$, where $A \in R_q^{1\times\ell}$ is a public matrix and $e_i \in R_q^{1\times\ell}$, and resets μ_c, ν_c to begin $(\omega + 1)$-th epoch.

To cope with perpetual leakage [44], each IS_i should update sk_i at the end of an epoch. We describe SkUpdate at the end of the ω-th epoch in Fig. 4. Notably, PartSign is actually a linearly homomorphic signature [41] in US3. Furthermore, the correctness of the equation $H(U_j^i) = \sum_{k=1}^{t-1} H(\alpha_k)^{i^k}$ can be found in Lemma 3 of [46] in US4. Finally, the $\mathsf{Verify}(id, pk_i, [U]_j^i, Sign_j)$ in US4 can be seen as Combine phase signature and message aggregation with 0. **Correctness**. According to Lemma 3, when $N \leq \frac{1}{4} log_2 \frac{L\cdot\ell\cdot n - \sigma\sqrt{n}}{\sigma\sqrt{n} - 1}$, the client C can recover authentication message sd_i in the login phase using the same password psw_c as in the registration phase and thus acquire an authentication token $AutToken$. Next, we show that the server-side key updates without changing the master private key after an epoch conversion. Accordingly, sd_i and $AutToken$ will not adjust. Notably, sd_i and $AutToken$ is related to msk. We show that even if sk_i changes after the epoch conversion, it does not change msk and therefore does not change sd_i and AutToken, which are generated by the same password.

Lemma 4. *At the end of the era, each server IS_i executes* SkUpdate. *IS_i can renews its private key sk_i without changing the master secret key msk.*

Proof. According to subsection 2.2, we know that $msk = \sum_{i=1}^{n} [S]_i^0 = \sum_{i=1}^{n} f_i(0)$. Suppose that $msk' = \sum_{i=1}^{n} f_i'(0)$. Since $f_i'(x) = f_i(x) + U_i(x)$, we have $msk' = \sum_{i=1}^{n} f_i'(0) = \sum_{i=1}^{n} f_i(0) + U_i(0) = \sum_{i=1}^{n} [s]_i^0 + [U]_i^0 = \sum_{i=1}^{n} [s]_i^0 + 0 = msk$. $\square$

Application. Our single sign-on scheme is general, i.e., it applies to all existing SSO scenarios. Notably, lattice passwords, despite their high computational efficiency (no power and bilinear pairing operations are required), however, require more storage overhead (because of the use of high-dimensional matrices). Therefore, our scheme is more suitable for devices with good storage resources.

5 Security Analysis

The registration algorithm can be regarded as a login algorithm sub-algorithm in our scheme. The password update can be considered as combining a login algorithm without issuing a token and a registration algorithm. Consequently, we mainly focus on the login phase. Furthermore, we are concerned about the security of the server-side secret key update algorithm. We analyze the security of our scheme from the following three theorems.

Theorem 1 (Quantum security). *The password-based threshold SSO authentication scheme in Fig. 4 is quantum security under appropriate parameter settings. For any PPT $\mathcal{A}$, the advantage holds that $Adv_{PTA,\mathcal{A}}^{\mathsf{Quantum}}(\kappa) \leq \varepsilon(\kappa)$.*

Proof. The most influential quantum attack algorithms are Shor's and Grover's. Quantum computers can efficiently solve large integer factorization and discrete logarithm problems with Shor's algorithm. Grover algorithm allows quantum computers to speed up the search for unstructured databases and hash collisions. To avoid the effects caused by both algorithms above, we construct the PTA scheme over lattices. The security of lattice-based hardness problem is reduced to the hardness of finding a relatively short vector in the lattice. $\mathcal{A}$ can execute the block-korkin-zolotarev (BKZ) algorithm [29] to find the short vectors in the n-dimensional lattice. Alkim et al. [50] proved that DRLWE can effectively resist quantum attacks (primal attack and dual attack) based on the BKZ algorithm.

The security of our TOPRF protocol is based on the DRLWE hardness assumption. From Fig. 2, the messages sent in the first step and the second part are encrypted by RLWE. If $\mathcal{A}$ can recover the random number through the first step or the server private key through the second part, they can solve the RLWE problem. From the proof of Lemma 3, it can be seen that $\mathcal{A}$ can solve the DRLWE problem if they can compute the server master private key or recover the user password by the message in the third step. Applying the "estimator" [51] with the quantum-security model [19] and $\kappa = 128, n = 768$, $log\ q = 23$, our TOPRF protocol can provide 138-bit security with $N \leq \frac{1}{4}log_2\frac{L\cdot\ell\cdot n-\sigma\sqrt{n}}{\sigma\sqrt{n}-1}$. According to definition 2, for any PPT $\mathcal{A}$, the advantage holds that: $Adv_{\mathcal{A}}^{\mathsf{DRLWE}}(\kappa) \leq \varepsilon(\kappa)$.

According to the unforgeability analysis in [39,41], if $\mathcal{A}$ can forge a signature, they can solve the SIS problem. Thus, the advantage of any PPT $\mathcal{A}$ holds that: $Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa) \leq \varepsilon(\kappa)$. According to the definition 11, the advantage of $\mathcal{A}$ to obtain sd_i by brutally cracking the symmetric encryption $E(z_i, sd_i)$ is $Adv_{\mathcal{A}}^{\mathsf{AES}}(\kappa) \leq \varepsilon(\kappa)$. In summary, our scheme can provide 128-bit security, and the advantage of any PPT $\mathcal{A}$ holds that:

$$Adv_{PTA,\mathcal{A}}^{\mathsf{Quantum}}(\kappa) = Adv_{\mathcal{A}}^{\mathsf{DRLWE}}(\kappa) + Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa) + Adv_{\mathcal{A}}^{\mathsf{AES}}(\kappa) \leq \varepsilon(\kappa).$$

□

Theorem 2 (Resist corrupt attacks). *Assuming that $\mathcal{A}$ cannot corrupt more than t servers in an epoch, $\mathcal{A}$ cannot obtain the master private key msk of IS.*

Proof. If $\mathcal{A}$ can calculate the corresponding msk through the obtained more than t private key in an epoch, $\mathcal{A}$ can forge the signature. Suppose $\mathcal{A}$ corrupted t servers and obtained t private keys in two consecutive epoch, denoted by $sk_1, ..., sk_{t'}, sk^*_{t'+1}, ..., sk^*_t$, $(a < t' < t < N)$. $\mathcal{A}$ calculates $\sum_{i=1}^{t'} \lambda_{ij} sk_i + \sum_{i=t'+1}^{t} \lambda_{ij} sk^*_i = \sum_{i=1}^{t'} \lambda_{ij} \sum_{j=1}^{N} [S]^i_j + \sum_{i=t'+1}^{t} \lambda_{ij} (\sum_{j=1}^{N} [S]^i_j + \sum_{j=1}^{N} [U]^i_j) = \sum_{i=1}^{t} \lambda_{ij} \sum_{j=1}^{N} [S]^i_j + \sum_{i=t'+1}^{t} \lambda_{ij} \sum_{j=1}^{N} [U]^i_j = msk + \sum_{i=t'+1}^{t} \lambda_{ij} \sum_{j=1}^{N} [U]^i_j$. Since the update key generated by 0-sharing satisfied the threshold security requirements, the adversary can obtain at most $t' < t$ update keys. In other words, there are $t - t'$ update keys that the adversary cannot get. Therefore, $\mathcal{A}$ cannot compute $msk = \sum_{i=1}^{t'} \lambda_{ij} sk_i + \sum_{i=t'+1}^{t} \lambda_{ij} sk^*_i - \sum_{i=t'+1}^{t} \lambda_{ij} \sum_{j=1}^{N} [U]^i_j$. □

Theorem 3 (Authentication). *In our scheme, let $\mathcal{A}$ can get pp and access client-side oracle and server-side oracle q_u times and q_s times, respectively. For any PPT $\mathcal{A}$, the advantage of disrupting authentication that $Adv^{\mathsf{Auth}}_{PTA,\mathcal{A}}(\kappa) \leq 2C' \cdot q_s^{s'}(\kappa) + (q_s + 2) Adv^{\mathsf{DLWE}}_{\mathcal{A}}(\kappa) + Adv^{\mathsf{SIS}}_{\mathcal{A}}(\kappa) + Adv^{\mathsf{AES}}_{\mathcal{A}}(\kappa) + \varepsilon(\kappa)$.*

Proof. Below, we provide the detailed analysis for the Theorem 3.
Experiment Exp_0^{Auth}. The simulator initializes sd_i, sid_j, L, and pp as defined in the real security experiment $Exp^{Auth}_{PTA,\mathcal{A}}(\kappa)$. $\mathcal{A}$ access oracle which is defined in subsection 3.2. that $Adv^{\mathsf{Auth}}_{PTA,\mathcal{A}}(\kappa) = Pr[succ_0^{Auth}]$.
Experiment Exp_1^{Auth}. This experiment is similar to Exp_0^{Auth} except that the random value s is ensure to be fresh in every session executed by the simulator through the oracles. We have $Pr[succ_1^{Auth}] - Pr[succ_0^{Auth}] = 0$.
Experiment Exp_2^{Auth}. This experiment is similar to Exp_1^{Auth} except that the simulator aborts if a value for x^* repeats in two different sessions of the protocol executed by the simulator through oracles. The password is hidden by random algorithms and noise, and the adversary cannot obtain password-related information unless the adversary can crack the LWE problem. Since the decision Diffie-Hellman-like problem is hardness, $\mathcal{A}$ cannot distinguish x^* or random value r. $Pr[succ_2^{Auth}] - Pr[succ_1^{Auth}] = Adv^{\mathsf{DLWE}}_{\mathcal{A}}(\kappa)$.
Experiment Exp_3^{Auth}. This experiment is similar to Exp_2^{Auth} except that server oracles receives the hidden value generated by the password chosen by the adversary x'. Since the decision Diffie-Hellman-like problem is hardness, $\mathcal{A}$ cannot distinguish $F_{msk}(x')$ and $F_{msk}(x)$. The password space follows the zipf law, the $Pr[succ_3^{Auth}] - Pr[succ_2^{Auth}] = C' \cdot q_s^{s'}(\kappa) + \varepsilon(\kappa)$.
Experiment Exp_4^{Auth}. This experiment is similar to Exp_3^{Auth} except that $\mathcal{A}$ chooses $(ID_{\mathcal{A}}, psw_{\mathcal{A}}, e_{\mathcal{A}})$ to queries $x^*_{k_i}$. Since the decision Diffie-Hellman-like problem is hardness, $\mathcal{A}$ cannot distinguish $x^*_{k_i}$ or $\boldsymbol{F}_q \leftarrow R_q^{1\times\ell} \cap \left(\frac{q}{p}\boldsymbol{y} + R^{1\times\ell}_{\leq\frac{q}{2p}}\right)$. In addition, each returned $x^*_{k_i}$ is generated by a different e_i. The probability of $\mathcal{A}$ gets client-specific key by query: $Pr[succ_4^{Auth}] - Pr[succ_3^{Auth}] \leq (q_s + 1) Adv^{\mathsf{DLWE}}_{\mathcal{A}}(\kappa) + C' \cdot q_s^{s'}(\kappa) + \varepsilon(\kappa)$.

Table 1: Security level of our scheme

	κ	m	σ	q	Hermit factor	Security level
PARMS I	95	512	22.93	4205569	1.004693	111-bits
PARMS II	128	768	9.73	8404993	1.003850	138-bits

Table 2: The computation overhead of each phase in our scheme

	PARMS I		PARMS II	
	Client side	Server side	Client side	Server side
Registration	0.627ms	0.043ms	1.582ms	0.109ms
Login	0.805ms	0.069ms	2.003ms	0.179ms
PswUpdate	0.585ms	0.089ms	1.455ms	0.223ms
SkUpdate	0	0.773ms	0	1.969ms

Experiment Exp_5^{Auth}. This experiment is similar to Exp_4^{Auth} except that $\mathcal{A}$ query $x^*_{k_i}$ and computes $F_{msk}(psw_{\mathcal{A}})$. $\mathcal{A}$ cannot obtain sd_i and z_i unless $\mathcal{A}$ can crack AES-256. $Pr[succ_5^{Auth}] - Pr[succ_4^{Auth}] = Adv_{\mathcal{A}}^{\mathsf{AES}}(\kappa)$.
Experiment Exp_6^{Auth}. This experiment is similar to Exp_5^{Auth} except that $\mathcal{A}$ corrupt $t' \leq t$servers and obtain their private keys. According to theorem 2, $\mathcal{A}$ cannot obtain an authentication token unless $\mathcal{A}$ can solve the SIS hardness problem and forge aggregate signature. $Pr[succ_6^{Auth}] - Pr[succ_5^{Auth}] = Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa)$.

In summary, for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage of disrupting authentication holds that:

$$Adv_{PTA,\mathcal{A}}^{\mathsf{Auth}}(\kappa) \leq 2C' \cdot q_s^{s'}(\kappa) + (q_s+2)Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) + Adv_{\mathcal{A}}^{\mathsf{SIS}}(\kappa) + Adv_{\mathcal{A}}^{\mathsf{AES}}(\kappa) + \varepsilon(\kappa).$$

□

6 Efficiency Analysis and Protocol Comparison

Our scheme is a single sign-on scheme constructed based on the lattice hardness problem. To meet the requirements of scheme quantum security, we employ the "lwe-estimator" scripts[6] to analyze the security level of our scheme under two different parameter settings. As shown in Table 1, our scheme can achieve 111-bit and 138-bit security, respectively.

Next, we estimate the computational cost through basic cryptographic operations. We write a multiplication operation as T_M, and an addition operation as T_A. T_H denotes once the hash operation, and T_S denote once Gaussian sampling. We use T_E to denote the encryption or decryption operation.

In the registration phase, the computation overhead of the client-side is $(2t+6)T_M + (2t+1)T_A + NT_H$, and a server's computation overhead is $T_M + T_A$.

[6] The "lwe-estimator" scripts are available at https://bitbucket.org/malb/lwe-estimator/src/master.

Table 3: Comparison the main OPRF protocol [17,48] with our work.

For L=64	Jarecki et al.[17]	Everspaugh et al. [48]	Our TOPRF
Client	1.431ms	5.227ms	0.641ms
Server	0.572ms	1.211ms	0.049ms

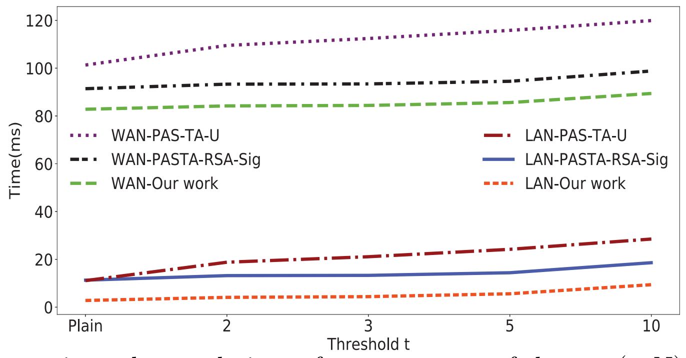


Fig. 5: Comparison the total time of current state-of-the-art (t, N) PTA-SSO schemes [12,20] with our work. Set $N = 10$, and the round-trip latency of WAN is 80ms. The Plain setting is the direct connection without the threshold setting.

In the login phase, the client and single server operations for $(3t + 7)T_M + 3tT_A + (t+1)T_H + 2tT_E$ and $T_M + T_A + T_S + 2T_E$ respectively. In the password update phase, the operation of the client is $(4t-2)T_M + 4tT_A + 2NT_H + 2NT_M$, and a server's operation is $2T_M + 2T_A + 2T_E$. The private key update phase requires no client participation and the computational overhead per server is $(Nt-N+2t+1)T_M+(Nt-2N+2t-1)T_A+(N+t-1)T_H+(2N-2)T_E+(N-1)T_S$.

Based on the above analysis, our C language reference implementation[7] is predicated on the q-TESLA 2.9 [47], and the measurement is obtained on a laptop with an AMD Ryzen 7-5800H running at 3.20 GHz. By Lemma 3, we set $L = 64$, $\ell = 256$, $N = 4$ and $t = 3$. The computational overhead of each phase in our scheme is shown in Table 2 for PARMS I and PARMS II. The time shown is calculated as the average of 1000 operations.

We compare the widely used PRF protocol [17,48] with our work (at Section **??**) in Table 3. Jarecki et al [17] first constructed the 2HashDH-based TOPRF in ACNS17, which is widely used in various PTA schemes [12,19,20]. The 2HashDH-based TOPRF requires more computational overhead since group-based cryptographic primitives need exponential power operations. Everspaugh et al. [48] proposed the Pythia PRF in USENIX15, which is employed in the PTA scheme of Zhang et al. [18]. The additional computational overhead of Pythia PRF arises from the power exponential and bilinear pair operations.

Form Table 4, we compare the standard Json Web Token [9], Agrawal et al. [12], Baum et al. [19], Zhang et al. [18], and Rawat et al. [20] with our scheme. Through comparative analysis, except for the standard Json Web Token, other schemes use multiple identity servers to authenticate the client and issue authentication tokens to prevent a single point of server failure. In addition, these protocols employ various PRF constructs to resist offline dictionary attacks.

[7] https://anonfiles.com/xaueI4kayb/QSSO_zip

Table 4: Comparison between Json Web Token [9], Agrawal et al. [12], Baum et al. [19], Zhang et al. [18], and Rawat et al. [20] and our work.

	Jones et al. [9]	Agrawal et al. [12]	Baum et al [19]	Zhang et al [18]	Rawat et al [20]	Our Work
Threshold	(1,1)	(t,N)	(N,N)	(t,N)	(t,N)	(t,N)
Server Corruption	–	static	adaptive	adaptive	static	adaptive
Server *sk*	$O(1)$	$O(C)$	$O(n)$	$O(G)$	$O(C)$	$O(G)$
Password Update	×	×	×	×	✓	✓
Security Proactive	×	×	✓	✓	×	✓
Offline Att. Resist.	×	✓	✓	✓	✓	✓
Online Att. Resist.	✓	✓	✓	✓	✓	✓
Quantum Secure	×	×	×	×	×	✓
Rounds	1	1	2	2	1	2
Efficiency Server	$1exe$	$2exe$	$4exe+1p$	0	$2exe$	0
Efficiency Client	0	$2exe$	$5exe$	$4p$	$2exe$	0

† Att. Resist.=Attack resistance; Server *sk* =Secret keys per server.
‡ C denotes the number of clients. G means the number of groups in the server.
∗ For efficiency we count the most expensive operations, i.e., exponentiations (denote by exe) and pairings (denot by p).

Baum et al. [19], Zhang et al. [18], and our scheme meets active security, which can resist perpetual leakage by updating the server-side key adaptively.

In terms of the number of protocol rounds in the login phase, compared with a round of the interaction of JSON Web Token [9], Agrawal et al. [12], and Rawat et al. [20], our protocol needs two rounds of interaction, but the increased interaction is meaningful. It can prevent malicious clients from obtaining authentication tokens that do not belong to them through impersonation attacks. In addition, only partial multiplication and addition are added to the increased number of rounds, and the protocol can still run efficiently. Finally, by comparing the exponential operation, and bilinear pair operation performed by the client and server in the login phase, we can get the conclusion consistent, that is, exponential and bilinear pair operation is not used in our scheme, which can make up for the computational overhead of symmetric encryption partly.

7 Conclusion and Future Work

In this paper, we propose a secure and effective password-based threshold single-sign-on authentication scheme over lattices. The proposal adopts multiple identity servers to authenticate the client, and issues authentication tokens to prevent a single point of server failure. It also supports servers to update the private key against perpetual leakage. Moreover, our scheme allows the client to update the password. Considering password guessing attacks in the identity authentication scheme, we propose a threshold oblivious pseudorandom function over lattices to resist offline password guessing attacks. We use a grouping structure to mitigate the harm caused by online password guessing attacks. We also employ UT [33] to construct a threshold homomorphism aggregation signature protocol to distribute authentication tokens. In addition, the design of the component in our scheme is based on lattice-based intractable problems against quantum attacks. Finally, we provide security proof to show that our scheme is secure and robust under various attacks and can deal with the adversary of quantum computing power. Our scheme is efficient and has comprehensive functions through efficiency analysis and comparison with five authentication schemes.

For future work, we will investigate how to reduce further the impact of noise accumulation to increase the allowable range of thresholds so that the threshold authentication system is scalable. In addition, it is meaningful to verify the validity of the computation during the interaction. However, zero-knowledge proofs are too heavy for authentication systems, and verification schemes similar to BLS [32] face the exact impact of noise accumulation during aggregation. We note that a series of prior art [49] uses the smooth projective hash function to generate strong session keys without further validation in authenticated key exchange schemes. Accordingly, a smooth projective hash function to construct a password-based threshold authentication scheme may be feasible.

References

1. J. Bonneau, C. Herley, P. Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In Proc. *IEEE S&P 2012*, pages 553–567.
2. J. Bonneau, C. Herley, P. van Oorschot, and F. Stajano. Passwords and the evolution of imperfect authentication. *Commun. ACM 2015*, 58(7):78–87.
3. A. Hanamsagar, S. Woo, C. Kanich and J. Mirkovic. Leveraging semantic transformation to investigate password habits and their causes. In Proc. *CHI 2018*.
4. A. Spadafora. Struggling with password overload? You're not alone. https://www.techradar.com/news/most-people-have-25-more-passwords-than-at-the-start-of-the-pandemic. Oct. 21, 2020.
5. D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang. Targeted online password guessing: An underestimated threat. In Proc. *ACM CCS 2016*, pages 1242–1254.
6. B. Pal, T. Daniel, R. Chatterjee and T. Ristenpart. Beyond credential stuffing: Password similarity models using neural networks. In Proc. *IEEE S&P 2019*.
7. A. Armando, R. Carbone, L. Compagna, J. Cuellar and L. Tobarra. Formal analysis of SAML 2.0 web browser single sign-on: breaking the SAML-based single sign-on for google apps. In Proc. *FMSE 2008*, pages 1–10.
8. B. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Commun. Mag. 1994*, 32(9):33–38.
9. M. Jones, J. Bradley and N. Sakimura. JSON Web Tokens. https://jwt.io/. Accessed on Dec. 15, 2021.
10. D. Wang and P. Wang. Offline dictionary attack on password authentication schemes using smart cards. In Proc. *ISC 2013*, pages 221–237.
11. J. Alwen, B. Chen, K. Pietrzak, L. Reyzin and S. Tessaro. Scrypt is maximally memory-hard. In Proc. *Eurocrypt 2017*, pages 33–62.
12. S. Agrawal, P. Miao, P. Mohassel and P. Mukherjee. PASTA: password-based threshold authentication. In Proc. *ACM CCS 2018*, pages 2042–2059.
13. P. MacKenzie, T. Shrimpton and M. Jakobsson. Threshold password-authenticated key exchange. In Proc. *CRYPTO 2002*, pages 385–400.

14. T. Rabin. A simplified approach to threshold and proactive RSA. In Proc. *CRYPTO 1998*, pages 89–104.
15. A. Bagherzandi, S. Jarecki, N. Saxena and Y. Lu. Password-protected secret sharing. In Proc. *ACM CCS 2011*, pages 433–444.
16. S. Jarecki, A. Kiayias and H. Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In Proc. *ASIACRYPT 2014*.
17. S. Jarecki, A. Kiayias, H. Krawczyk and J. Xu. TOPPSS: cost-minimal password-protected secret sharing based on threshold OPRF. In Proc. *ACNS 2017*.
18. Y. Zhang, C. Xu, H. Li, K. Yang, N. Cheng and X. Shen. PROTECT: efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage. *IEEE Trans. Mob. Comput. 2020*, 20(6): 2297–2312.
19. C. Baum, T. Frederiksen, J. Hesse, A. Lehmann and A. Yanai. PESTO: proactively secure distributed single sign-on, or how to trust a hacked server. In Proc. *EuroS&P 2020*, pages 587–606.
20. R. Rawat and M. Jhanwar. PAS-TA-U: PASsword-Based Threshold Authentication with Password Update. In Proc. *SPACE 2020*, pages 25–45.
21. T. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe and J. OBrien. Quantum computers. *Nature 2010*, 464(7285): 45-53.
22. V. Mavroeidis, K. Vishi, M. Zych and A. Jøsang. The impact of quantum computing on present cryptography. *Int. J. Adv. Comput. Sci. Appl. 2018*, 9(3): 405-414. *IEEE Trans. Mob. Comput. 2020*, 20(6): 2297–2312.
23. G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y. Liu, C. Miller, D. Moody, R. Peralta and others. Status report on the first round of the NIST post-quantum cryptography standardization process. https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf. 2019.
24. G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, J. Kelsey, Y. Liu, C. Miller, D. Moody, R. Peralta and others. Status report on the second round of the NIST post-quantum cryptography standardization process. Status report on the second round of the NIST post-quantum cryptography standardization process. NIST,Tech. Rep., July 2020.
25. J. Ding, S. Alsayigh, J. Lancrenon, R. Saraswathy and M. Snook. Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In Proc. *CT-RSA 2017*, pages 183–204.
26. Z. Li and D. Wang. Two-round PAKE protocol over lattices without NIZK. In Proc. *INSCRYPT 2018*, pages 138–159.
27. J. W. Bos, C. Costello, M. Naehrig and D. Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In Proc. *IEEE S&P 2015*, pages 553–570.
28. J. Zhang, Z. Zhang, J. Ding, M. Snook and O. Dagdelen. Authenticated key exchange from ideal lattices. In Proc. *EUROCRYPT 2015*, pages 719–751.
29. C. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program. 1994*, 66(1): 181–199.
30. L. Grover. A fast quantum mechanical algorithm for database search. In Proc. *STOC 1996*, pages 212–219.
31. G. Alagic, D. Apon, D. Cooper, Q. Dang, and others. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. Gaithersburg, MD: National Institute of Standards and Technology, 2022.
32. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In Proc. *ASIACRYPT 2001*, pages 514–532.

33. D. Boneh, R. Gennaro, S. Goldfeder, A. Jain, S. Kim, P. Rasmussen and A. Sahai. Threshold cryptosystems from threshold fully homomorphic encryption. In Proc. *CRYPTO 2018*, pages 565–596.
34. M. Albrecht, A. Davidson, A. Deo and N. Smart. Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. In Proc. *PKC 2021*, pages 261–289.
35. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In Proc. *EUROCRYPT 2000*, pages 139–155.
36. V. Lyubashevsky, C. Peikert and O. Regev. On ideal lattices and learning with errors over rings. *J. ACM 2013*, 60(6):1–35.
37. A. Shamir. How to share a secret. *ACM Commun. 1979*, 22(11): 612–613.
38. R. Bendlin, S. Krehbiel and C. Peikert. How to share a lattice trapdoor: threshold protocols for signatures and (H) IBE. In Proc. *ACNS 2013*, pages 218–236.
39. Z. Jing. An efficient homomorphic aggregate signature scheme based on lattice. *Math. Probl. Eng. 2014*, 2014(1): 1–9.
40. S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris and W. Hoeteck. Functional encryption for threshold functions (or fuzzy IBE) from lattices. In Proc. *PKC 2012*, pages 280–297.
41. F. Wang, Y. Hu and B. Wang. Lattice-based linearly homomorphic signature scheme over binary field. *Sci. China Inf. Sci. 2013*, 56(11):1–9.
42. A. Banerjee and C. Peikert. New and improved key-homomorphic pseudorandom functions. In Proc. *CRYPTO 2014*, pages 353–370.
43. D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner and M. Zhandry. Random oracles in a quantum world. In Proc. *ASIACRYPT 2011*, pages 41–69.
44. A. Herzberg, S. Jarecki, H. Krawczyk and M. Yung. Proactive secret sharing or: How to cope with perpetual leakage. In Proc. *CRYPTO 1995*, pages 339–352.
45. D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian. Zipf's law in passwords. *IEEE Trans. Inf. Foren. Sec. 2017*, 12(11): 2776–2791.
46. B. Rajabi and Z. Eslami. A verifiable threshold secret sharing scheme based on lattices. *Inf. Sci. 2019*, 501:655–661.
47. E. Alkim, P. Barreto, N. Bindel, J. Krämer, P. Longa and J. Ricardini. The lattice-based digital signature scheme qTESLA. In Proc. *ACNS 2020*, pages 441–460.
48. A. Everspaugh, R. Chaterjee, S. Scott, A. Juels and T. Ristenpart. The Pythia PRF Service. In Proc. *USENIX SEC 2015*. pages 547–562.
49. Z. Li, D. Wang and E. Morais. Quantum-safe round-optimal password authentication for mobile devices. *IEEE Trans. Depend. Secur. Comput. 2020*, 19(3).
50. E. Alkim, L. Ducas, T. Poppelmann, and P. Schwabe. Post-quantum key exchange a new hope. In Proc. *USENIX SEC 2016*, pages 327–343.
51. M.R. Albrecht, R. Player and S. Scott. On the concrete hardness of learning with errors. *J. CRYPTOL 2015*, 9(3): 169–203.