Accepted by IEEE transactions on Services Computing, 2023, Doi: 10.1109/TSC.2023.3331000



# QPause: Quantum-Resistant Password-Protected Data Outsourcing for Cloud Storage

Jingwei Jiang, Ding Wang, and Guoyin Zhang

***Abstract*—Cloud storage provides an efficient and convenient way to manage data, but it also poses significant challenges to data security. The central issue with cloud storage is to ensure the ability of the data owner to control and manage the outsourced data. The password-protected secret sharing (PPSS) integrates password authentication and secret sharing to offer a fresh approach to secure private data. Users can share the risk of device corruption with a well-designed PPSS scheme and manage outsourced data with only human-memorizable passwords. To the best of our knowledge, none of the existing PPSS schemes can resist security threats in the post-quantum era, and there is an urgent need to design quantum-resistant solutions. However, post-quantum cryptography varies significantly from traditional cryptography, and it is challenging to design a quantum-resistant password-protected secret-sharing scheme for cloud storage.**

**In this work, we take the first substantial step towards this challenge by proposing QPause, a quantum-resistant password-protected data outsourcing scheme for cloud storage. We first design a basic quantum-resistant PPSS scheme based on the lattice secure against semi-honest adversaries with a secure channel. On this foundation, we propose a quantum-resistant round-optimal password-protected data outsourcing scheme against strong adversaries. In addition, we formally prove that our scheme is secure and robust under various attacks against adversaries with quantum computing capabilities. The comparison results show that our new scheme outperforms its foremost counterparts.**

***Index Terms*—Quantum security, Lattice, Password protected, Secret sharing, Cloud storage.**

## I. Introduction

THE wide use of fifth Generation (5G), edge computing, and Internet of Things technologies has produced an enormous amount of data. According to IDC's prediction, the global data will grow to 175 zeta bytes by 2025 [1]. Data storage is becoming increasingly critical. Cloud storage can effectively integrate and utilize the traditional scattered and isolated data information. Hence, the in-depth value contained in the data can play an influential role. Outsourced data to the cloud servers relieves the users from complex local storage management and maintenance. However, it may make users lose control of their data and bring significant challenges to data security. These days it is no news to hear that user data are breached in attacks targeting cloud service providers, such as [2], [3]. With attacks targeting cloud service providers happening on a daily basis [2], it is essential to protect the privacy and security of outsourced data kept in cloud storage.

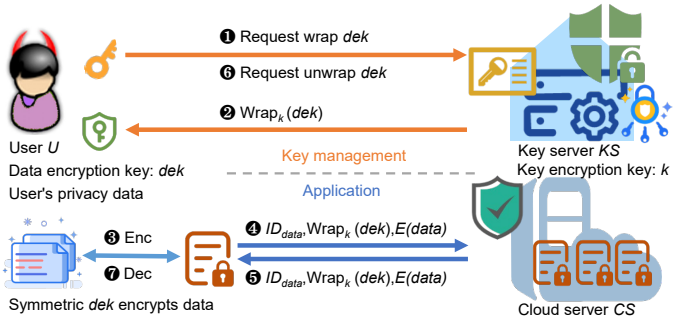


Fig. 1. An exemplary overview of a symmetric encryption system for cloud storage. The user $U$ selects a data encrypting key $dek$ and ❶ sends $dek$ to the key server $KS$. ❷ $KS$ wraps $dek$ with the key encryption key $k$ and returns to $U$. ❸ $U$ uses $dek$ to encrypt the data, and ❹ uploads the data ID $ID_{data}$, the wrapping value $\text{Wrap}_k(dek)$ and the encrypted data $E(data)$ to the cloud server $CS$. When $U$ needs data, she ❺ downloads $\{ID_{data}, \text{Wrap}_k(dek), E(data)\}$ from $CS$, and ❻ sends the wrapping value to $KS$ for unwrapping. ❼ $U$ can decrypt $E(data)$ with $dek$ to get the data.

Encryption of outsourced data is the most direct technique to prevent data leakage [4]. In practice, cloud servers are not always honest, and it is not secure for users to manage cloud-based encryption data through authentication only [5]. Furthermore, users expect to gain more control over the data they own [6], so allowing users to encrypt data locally and then upload the ciphertext is a more secure way to store cloud data [7]. However, the management of data encryption keys causes additional storage overhead to the user.

An alternative solution is to introduce an independent third-party key management system (KMS) (e.g., Google Cloud [9]) that uses the wrap-unwrap approach to support users in managing large amounts of symmetric data encryption keys (dek) to encrypt outsourced data as shown in Fig. 1. However, the key encapsulation mechanism has a significant potential security vulnerability. In the first step of Fig. 1, *dek* is sent to the KMS in plaintext, ensuing in *dek* being exposed to a malicious KMS and easily obtained by adversaries through man-in-the-middle attacks. Additionally, updating $k$ requires unwrapping and re-wrapping all *dek*.

Jarecki et al. proposed the oblivious key management system (OKMS) [10]. The OKMS employs an oblivious way (oblivious pseudorandom function [11]) to address key compromises arising from a remote KMS. (Unlike updatable encryption [12], [13] that solves the security problem of key management locally on the client side). Moreover, OKMS is extended to updatable OKMS (UOKMS) through key rotation to ensure the forward security of the system. Despite the great benefits that UOKMS brings, the inadequacy of an effective authentication mechanism can make UOKMS vulnerable to

J.W. Jiang, and G.Y. Zhang, are with College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China. J.W. Jiang is also with Henan Key Laboratory of Network Cryptography Technology.

D. Wang is with Key Laboratory of Data and Intelligent System Security (Nankai University), Ministry of Education, Tianjin 300350, China, and also with Henan Key Laboratory of Network Cryptography Technology. (E-mail: wangding@nankai.edu.cn.). Ding Wang is the corresponding author.

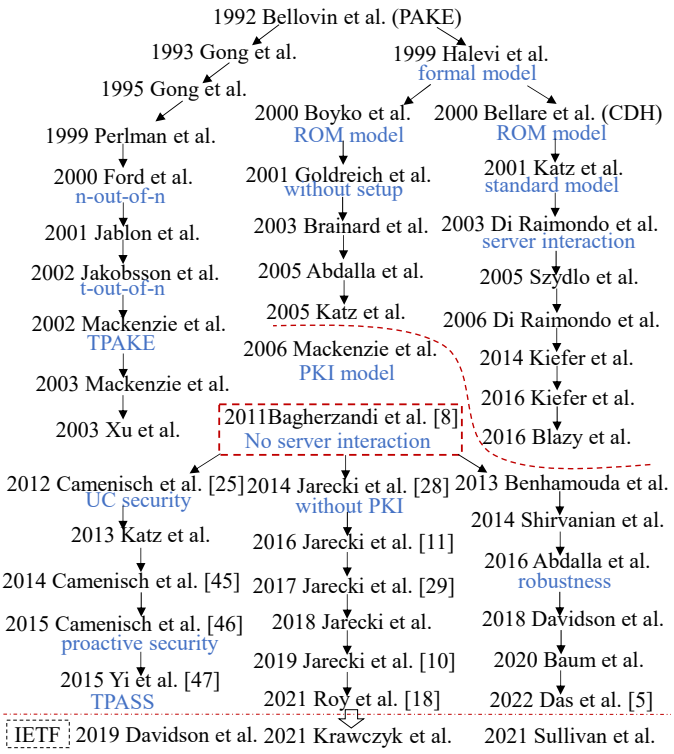


Fig. 2. A brief history of password-protected secret sharing (PPSS). The seminal contributions of some works are shown in corresponding blue words. Bagherzandi et al. [8] propose the first formal PPSS, which is marked in red.

impersonation attacks. Specifically, the adversary impersonates the target user to download encrypted data from the cloud server, and interacts with the KMS to obtain the data decryption key. Finally, the adversary extracts the private data. Hence, both the cloud server and KMS need to authenticate the user to confirm their access to resist impersonation attacks. Both the cloud storage provider and KMS need to authenticate the user requesting the service to confirm their access. Password-protected secret sharing (PPSS) provides an elegant solution for implicit authentication while achieving key management.

In this work, we propose a lattice-based PPSS under the PPSS secure model [8] to resist various attacks (e.g., offline password guessing attack [14], corrupt attack [15], signal leakage attacks [16], and key mismatch attack [17]) from both quantum computing and classical computing adversaries. We solve the challenge of leaking the secret shares of server-side keys during distributed decryption in Roy et al.'s lattice-based PPSS [18] and implement a quantum-resistant data outsourcing scheme, named QPause. Under a stricter password distribution model (i.e., Zipf-distribution [19], see Fig. 4), our formal proof demonstrates that QPause is secure and robust.

## II. Related Works

Password-based authentication remains the most frequently used authentication mechanism in various web systems [20]. Although passwords have inherent defects in both security and usability, passwords are still stubbornly surviving among various alternative authentication schemes and are surging with almost every new network service [21]. There is a growing consensus that password-based authentication is likely to retain its status for the foreseeable future [22]. However, it is unrealistic to rely solely on passwords to achieve outsourced data security. Because the data encryption key is generally a series of random long numbers with high entropy, while there are 20-22 bits of entropy on average for a human-memorizable password [23], [24]. How can users authenticate and manage outsourced data by employing a password only?

Bagherzandi et al. [8] propose password-protected secret sharing (PPSS), which provides an elegant solution to this problem. With the PPSS scheme, the user $U$ can encrypt data locally and upload them to the cloud server $CS$. Then, $U$ data encryption key is shared with $N$ key servers $KS$. After downloading the data cipher from $CS$, $U$ only needs to input a correct human-memorizable password to retrieve data by interacting with at least $t+1$ responding to honest $KS$.

No password-related files need to be stored on the key server side, so even if $t$ key servers are corrupted, the scheme's security remains the same as that of password-based authentication schemes. In addition, PPSS schemes can employ non-interactive zero-knowledge proofs (NIZK) [8], [25], [18] to meet security in malicious-resistant environments.

Bagherzandi et al. [8] spark a number of studies in the field of PPSS [11], [26], [27], [25], [28], [29]. For a more concrete grasp, we summarize the history of PPSS in Fig. 2. Note that a series of other important schemes about PPSS cannot be incorporated into the Fig. 2 only because of space constraints.

To the best of our knowledge, none of those mentioned schemes in Fig.2 can resist security threats in the post-quantum world: 1) All but one scheme (i.e., Roy et al.'s [18]) are built on the hardness of traditional cryptography assumptions (e.g., large integer decomposition, discrete logarithms, elliptic curves), which is vulnerable after the advent of quantum computers (which are known to perform Shor algorithms [30] to solve traditional hardness problems efficiently); 2) Roy et al.'s PPSS scheme [18] is based on learning with errors (LWE) and attempted to resist the quantum attacker, but as we show in Section III, it is vulnerable to the man-in-the-middle attacks and cannot achieve its goal (i.e., protecting the secret key).

**Problem Statement.** With the recent advancements in quantum computing [31], [32], standards organizations (e.g., IETF, IEEE, and NIST) are preparing solutions in the post-quantum age. Lattice-based schemes are regarded as the most promising general-purpose algorithms for public key encryption by NIST [33]. Dozens of quantum-resistant password-based protocols [34], [35] have been proposed over lattices. However, to the best of our knowledge, *there is no quantum-resistant password-protected data outsourcing scheme* (for cloud storage). The main goal of our solution is to answer the question:

> *Is it possible to construct a quantum resistant password protected data outsourcing scheme over lattices to satisfy that only the user who knows the password can outsource and retrieve data?*

Our answer to the above question is affirmative.

## III. Overview of Our Technique

**Motivations.** According to the discussion above, we have established a primary research direction, i.e., we first construct a PPSS scheme, and then design a password-protected data

outsourcing for cloud storage on this basis. Since Bagherzandi et al. [8] formally proposed the first PPSS scheme, considerable efforts have been devoted to developing secure and efficient PPSS schemes. Yet, no secure and effective quantum resistance scheme has been formally proposed.

Fundamentally, the design of a secure cryptographic scheme needs to rely on the intractable problem, which no known algorithm can solve in polynomial time. But Shor's algorithm [30] can efficiently solve the intractable problems relied on by traditional cryptography via quantum computers. In the coming quantum era, it is necessary to design a quantum secure password-protected data outsourcing scheme for cloud storage. However, we cannot directly translate existing schemes into quantum resistant schemes because of a series of challenges inherent in the computational methods and security objectives.

On the one hand, we discuss the challenges in the computational methods. First, the construction of Bagherzandi et al. [8] employed ElGamal encryption, which is vulnerable to adversaries of quantum computing power. We need to redesign the basic PPSS using a quantum-resistant homomorphic encryption algorithm and a threshold decryption algorithm over lattices. However, the lattice-based homomorphic group encryption [36], [37] provide no support for password re-randomization. Roy et al. [18] devised an ingenious construction with fully homomorphic encryption (FHE) [38].

However, they try to thresholdize FHE decryption in the decryption phase by directly applying $t$-out-of-$N$ Shamir secret sharing to $sk$. The user computes the Lagrange coefficients $\lambda_i$ for the subset $S \subseteq \{1, ..., N\}$ of size $t$, and recombine the shares as $\sum_{i \in S} \lambda_i \cdot \langle \mathbf{ct}, \mathbf{sk}_i \rangle = \langle \mathbf{ct}, \sum_{i \in S} \lambda_i \cdot \mathbf{sk}_i \rangle = \langle \mathbf{ct}, \mathbf{sk} \rangle$, where $\mathbf{ct}$ denotes the ciphertext, and $\mathbf{sk}_i$ denotes the share of the secret key $\mathbf{sk}$. However, Boneh et al. [39] pointed out that each partial decryption operation leaks information about $\mathbf{sk}_i$ by publishing the inner product of $\mathbf{sk}_i$ with $\mathbf{ct}$. Concretely, the adversary $\mathcal{A}$ can launch man-in-the-middle attacks to obtain the results of partial decryption. Then, $\mathcal{A}$ can calculate the server-side secret keys by the Gaussian elimination method.

Second, the threshold decryption algorithm of the PPSS scheme is critical for data retrieval. However, the aggregation process of the lattice threshold algorithm can amplify the noise, which leads to decryption failure. Therefore, when designing a lattice-based PPSS scheme, it is necessary to ensure that the aggregation coefficients are low-norm and the coefficients remain integer after aggregation [40]. Third, we extend the basic PPSS scheme to a password-protected data outsourcing scheme for cloud storage. Based on resisting quantum attacks [30], [41], offline dictionary attacks [42], and corruption attacks [43], [15], we should fully consider the existence of malicious adversaries (including malicious users and malicious key servers) in the real world, i.e., adversaries who actively tamper with messages (e.g., tampering with random numbers and re-randomizing encrypted passwords in cloud servers) and malicious computations (incorrect execution of protocols leading to data recovery failure).

On the other hand, we discuss the challenges in security objectives. In the PPSS scheme, adversaries perform corruption attacks on key servers to obtain information about data encryption keys. Under the assumption of the $(t, N)$ threshold, adversaries can corrupt at most $t$ key servers. At this point, the level of this protection is as expected of the password-authenticated protocol. It is commonly assumed in PPSS [8] that the selection of passwords is uniformly distributed, and the probability that adversaries obtain outsourced data is at most $q_{send}/|D|$, where $q_{send}$ is the number of most online attacks, and $|D|$ is the size of the password dictionary. Recent research results [19] suggest that password selection in fact follows Zipf-distribution, i.e. the adversary's advantage as $C' \cdot q_{send}^{s'}(\kappa) + \varepsilon(\kappa)$ for the Zipf parameters $C'$ and $s'$. Wang et al. [19], [14] showed that the advantages of the adversary are underestimated in the uniform model. The impact of password distribution assumptions should be fully considered.

**Contributions.** We take the first substantial step toward a quantum-resistant scheme by proposing the **Q**uantum resistant **Pa**ssword-protected data o**u**tsourcing scheme for cloud **s**torag**e**, named QPause. Our construction relies on the learning with errors (LWE) for which efficient quantum algorithms are not known. Our construction starts with the general scheme of PPSS [8] but employs quantum-secure cryptographic primitives. In summary, our contributions are three-fold:

- **A PPSS over LWE**. We first design a basic quantum-resistant PPSS over lattices secure against semi-honest adversaries with secure channels. We devise a method for a re-randomization password applicable to lattice-based fully homomorphic encryption. It ensures that the outsourced data remains masked with a re-randomization password, which cannot be distinguished from random numbers. Besides, we employ clear out their denominators [40] to get low-norm interpolation coefficients. It reduces the impact of noise on the decryption phase. Finally, we add well-structured noise to the partial decryption without affecting the final decryption. It ensures that the inner product leaks nothing about the secret share.
- **QPause for cloud stotage**. We formally propose a quantum-resistant password-protected data outsourcing scheme against malicious adversaries, named QPause. The QPause requires no secure channels and has a round-optimal. We generate temporary public-secret key pairs in the QPause to encrypt the transmitted messages. It can prevent eavesdropping attacks. Moreover, we employ promises and simulation sound non-interaction zero-knowledge proof (SS-NIZK) [44] to ensure the freshness and validity of computation. The SS-NIZK can instantiate the universal thresholdizer to eliminate the impact of reduced interpolation coefficients, i.e., the size of the ciphertext now depends on the number of servers. It can make our QPause achieve compactness and robustness.
- **Security and performance.** We evaluate the security of QPause under two parameter settings, and make a rigorous security proof under the Zipf model [19] and the PPSS secure model [8]. Besides, we demonstrate the compactness, robustness, and soundness of QPause. Performance evaluations show that our scheme outperforms the state-of-the-art PPSS [18] in terms of computation and communication in the recovery phase. Comparison results show that our scheme is superior to its counterparts [5], [8], [10], [11], [25], [28], [29], [18], [45], [46], [47].

## IV. PRELIMINARIES

**Notations.** We use $\kappa$ to denote the security parameter. Let $\mathbb{Z}$ and $\mathbb{R}$ denote the set of all integers and the set of real numbers, respectively. For any integer $q$, let $\mathbb{Z}_q$ denote the ring of integer mod $q$. Let lower-case bold $\mathbf{x}$ letter denote vectors, and upper-case bold letter $\mathbf{A}$ represent matrices. We use $x \leftarrow \mathcal{D}$ to denote the sampling of $x$ according to distribution $\mathcal{D}$ and $x \leftarrow S$ for a finite set $S$ to indicate sample uniformly at random from $S$.

### A. Lattices, LWE, and Gaussian Sampling

**Definition 1** ([37]). *Let $\Lambda_q(\mathbf{A}) = \{\mathbf{As} \mid \mathbf{s} \in \mathbb{Z}_q^n\}$ denotes an $m$-dimensional lattice with the basis $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ for $m \geq n \cdot log\ q$, and the determinant of $\Lambda$ is $det(\Lambda) = \sqrt{det(\mathbf{B}^T\mathbf{B})}$.*

**Definition 2** (**Gaussian distributions** [36]). *For a standard deviation $\sigma > 0$, define the discrete Gaussian distribution over an integer lattice $\Lambda \subseteq \mathbb{Z}^m$ centred at $\mathbf{c} \in \mathbb{R}^n$ with parameter $\sigma$ to be: $D_{\mathcal{R},\sigma}(\mathbf{z}) = \rho_{\mathbf{c},\sigma}(\mathbf{x})/\rho_{\mathbf{c},\sigma}(\Lambda)$, where $\mathbf{x} \in \Lambda$, $\rho_{\mathbf{c},\sigma}(\mathbf{x}) = e^{\pi||\mathbf{x}-\mathbf{c}||^2/\sigma^2}$, and $\rho_{\mathbf{c},\sigma}(\Lambda) = \sum_{\mathbf{x}\in\Lambda} \rho_{\mathbf{c},\sigma}\mathbf{x}$.*

**Definition 3** (Decision-$\mathsf{LWE}_{n,q,\chi,m}$ [37]). *For a prime integer $q$, integers $m, n > 0$, and a noise distribution $\mathcal{X}$ over $\mathbb{Z}_q$, sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{m\times n}, \mathbf{s} \leftarrow \mathbb{Z}_q^{n\times 1}, \mathbf{e} \leftarrow \chi^{m\times 1}, \mathbf{b} \leftarrow \mathbb{Z}_q^m$. The DLWE$_{n,q,\chi,m}$ problem is to distinguish between $(\mathbf{A}, \mathbf{A}\cdot\mathbf{s} + \mathbf{e}\ mod\ q) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times 1}$ and $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times 1}$. For any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, the two distinct distributions are computationally indistinguishable, i.e., $Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) = |Pr[\mathcal{A}(q, m, n, \mathcal{X}, \mathbf{A}, \mathbf{s})] - Pr[\mathcal{A}(q, m, n, \mathcal{X}, \mathbf{A}, \mathbf{b})]| \leq \varepsilon(\kappa)$.*

### B. Shamir Secret Sharing

The Shamir secret sharing (SS) [43] allows the user to divide the secret $s$ into $N$ pieces in such a way that $s$ can be restored from any $t+1$ pieces, but even complete knowledge of $t$ pieces reveals absolutely no information about $s$. We provide an introduction to the basic notations and terms in SS. Our definitions are adapted from Boneh et al. [39].

**Definition 4.** *Let $P = \{P_1, ..., P_N\}$ be a set of parties, $\mathbb{A}$ be a class of efficient access structures on $P$, and $\mathbb{S}$ be the class of threshold access structures on $P$. For the secret space $\mathcal{S} = \mathbb{Z}_p$, where $p$ is a prime. A Shamir secret sharing scheme must contain the following two polynomial algorithms:*

- *$(\mathbf{s}_1, ..., \mathbf{s}_N) \leftarrow \mathsf{SS.Share}(s, \mathbb{A})$: On input a secret $s_0 = s \in \mathcal{S}$ and an access structure $\mathbb{A}$, the $\mathsf{SS.Share}$ algorithm outputs a set of shares $\{\mathbf{s}_1, ..., \mathbf{s}_N\}$ for each parties.*
- *$s \leftarrow \mathsf{SS.Combine}(\{\mathbf{s}_i\}_{i\in\mathbb{S}})$: For any $i, j \in [1, N]$ and the size of $\mathbb{S}$ is $t+1$, the Lagrange coefficients $\lambda_{i,j}^{\mathbb{S}} = \prod_{i\neq j}^{t} \frac{-I_i}{(I_j - I_i)}$, such that $s_0 = \sum_{i\in\mathbb{S}} \lambda_{i,j}^{\mathbb{S}} \cdot s_i$. The $\mathsf{SS.Combine}$ outputs $s = s_0$.*

### C. Fully Homomorphic Encryption

We briefly recall the encryption scheme of GSW construction [38]. For $\mathbf{x} \in \mathbb{Z}_q^m$, $\ell = \lfloor log\ q + 1 \rfloor$, and $n = m\ell$. We write $\mathsf{BitDecomp}(\mathbf{x})$ as the n-dimensional vector $\mathbf{x}' = (x_{1,0}, ..., x_{1,\ell-1}, ..., x_{m,0}, ..., x_{m,\ell-1})$, where $x_{i,j}$ is the $j$-th bit in the binary representation of $x_i$. The inverse of $\mathsf{BitDecomp}$ is represented by $\mathsf{BitDecomp}^{-1}$ satisfying: $\mathsf{BitDecomp}^{-1}(\mathbf{x}') = (\sum_{j=0}^{\ell-1} 2^j \cdot x_{1,j}, ..., \sum_{j=0}^{\ell-1} 2^j \cdot x_{m,j})$. We define $\mathsf{Flatten}(\mathbf{x}') = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\mathbf{x}'))$. Let $\mathsf{Powesrof2}(\mathbf{x}) = (x_1, 2x_1, ..., 2^{\ell-1}x_1, ..., x_m, 2x_m, ..., 2^{\ell-1}x_m)$.

**Definition 5** ([38]). *For $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_q^m$ and any $n$-dimensional vector $\mathbf{x}'$, the following three equations hold:*

- *$\langle \mathsf{BitDecomp}(\mathbf{x}), \mathsf{Powersof2}(\mathbf{y}) \rangle = \langle \mathbf{x}, \mathbf{y} \rangle$,*
- *$\langle \mathbf{x}', \mathsf{Powersof2}(\mathbf{y}) \rangle = \langle \mathsf{BitDecomp}^{-1}(\mathbf{x}'), \mathbf{y} \rangle$,*
- *$\langle \mathsf{BitDecomp}^{-1}(\mathbf{x}'), \mathbf{y} \rangle = \langle \mathsf{Flatten}(\mathbf{x}'), \mathsf{Powersof2}(\mathbf{y}) \rangle$.*

**Definition 6.** *According to Definition 3, let $\mathcal{X}$ is a noise distribution, $n \in \mathbb{Z}^+$, $q = poly(n)$ is power of 2, and $m = \Theta(nlog\ q)$. $\mathcal{N} = (n+1) \cdot (\lfloor log\ q \rfloor + 1)$. For $pp = (n, q, \mathcal{X}, m)$ and $\mathcal{N}$, a fully homomorphic encryption scheme of GSW contain the following three polynomial algorithms:*

- *$(\mathbf{pk}, \mathbf{sk}) \leftarrow \mathsf{KeyGen}(pp)$: On input the $pp$, sample $\mathbf{k} \leftarrow \mathbb{Z}_q^n$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{m\times n}$, and $\mathbf{e} \leftarrow \mathcal{X}^m$. Compute $\mathbf{a} = \mathbf{A}\cdot\mathbf{k} + \mathbf{e}$ and set $\mathbf{s} = (1| - \mathbf{k}) \in \mathbb{Z}_q^{n+1}$. Output $(\mathbf{pk}, \mathbf{sk})$, where $\mathbf{pk} = [\mathbf{a}|\mathbf{A}] \in \mathbb{Z}_q^{m\times(n+1)}$ and $\mathbf{sk} = \mathsf{Powersof2}(\mathbf{s})$.*
- *$\mathbf{C} \leftarrow \mathsf{Enc}(pp, \mathbf{pk}, M)$: On input a message $M \in \mathbb{Z}_q$, sample $\mathbf{R} \leftarrow \{0,1\}^{\mathcal{N}\times m}$ and compute $\mathbf{C} = \mathsf{Flatten}(M \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}\cdot\mathbf{A})) \in \mathbb{Z}_q^{\mathcal{N}\times\mathcal{N}}$.*
- *$M \leftarrow \mathsf{Dec}(pp, sk, \mathbf{C})$: On input a ciphertext $\mathbf{C}$ and $\mathbf{sk}$, $\{1, 2, ..., 2^{\ell-1}\}$ are the first $\ell$ coefficients of $\mathbf{sk}$ and $\mathbf{sk}_i = 2^i \in (q/4, q/2]$. Compute $\mathbf{C}_i \cdot \mathbf{sk}/\mathbf{sk}_i$, where $\mathbf{C}_i$ be the $i$-th row of $\mathbf{C}$, and rounding the noise to get $\mathbf{M}$.*

According to Definition 4 and Definition 6, we can construct the threshold fully homomorphic encryption (TFHE). The TFHE allows the decryption key to be split into shares, such that any class of access structures can be combined into a complete decryption of a given ciphertext. Our definitions are adapted from Boneh et al. [39] as follows:

**Definition 7.** *(TFHE) Let $P = \{P_1, ..., P_N\}$ be a set of parties and let $\mathbb{S}$ be a class of efficient access structure on $P$. A threshold fully homomorphic encryption scheme must contain the following five polynomial algorithms:*

- *$(\mathbf{pk}, \mathbf{sk}_1, ..., \mathbf{sk}_N) \leftarrow \mathsf{TFHE.Setup}(1^\kappa, 1^d, \mathbb{A})$: On input the security parameter $\kappa$, a depth bound $d$, and an access structure $\mathbb{A}$, the setup algorithm outputs a public key $\mathbf{pk}$, and a set of secret key shares $\mathbf{sk}_1, ..., \mathbf{sk}_N$.*
- *$ct \leftarrow \mathsf{TFHE.Enc}(\mathbf{pk}, m)$: On input a public key $\mathbf{pk}$, and a single bit plaintext $m \in \{0,1\}$, the encryption algorithm outputs a ciphertext $\mathbf{ct}$.*
- *$\hat{ct} \leftarrow \mathsf{TFHE.Eval}(\mathbf{pk}, C, \mathbf{ct}_1, ..., \mathbf{ct}_k)$: On input a public key $pk$, circuit $C : \{0,1\}^k \rightarrow \{0,1\}$ of depth at most $d$, and a set of ciphertexts $\mathbf{ct}_1, ..., \mathbf{ct}_k$, the evaluation algorithm outputs a cipehrtext $\hat{\mathbf{ct}}$.*
- *$\mathbf{p}_i \leftarrow \mathsf{TFHE.PartDec}(\mathbf{ct}, \mathbf{sk}_i)$: On input a $\mathbf{ct}$ and a secret key share $\mathbf{sk}_i$, the partial decryption algorithm outputs a partial decryption $\mathbf{p}_i$ related to the party $P_i$.*
- *$\hat{m} \leftarrow \mathsf{TFHE.FinDec}(\mathbf{sk}, \mathbf{B})$: On input a secret key $\mathbf{sk}$, and a set $\mathbf{B} = \{\mathbf{p}_i\}_{i\in\mathsf{S}}$ for $\mathsf{S} \subseteq \{P_1, ..., P_N\}$, the final decryption algorithm outputs a plaintext $\hat{m} \in \{0, 1, \perp\}$*

**Definition 8** (Compactness). *Let $poly(\cdot)$ denote polynomial. For the security parameter $\kappa$, depth bound $d$, circuit $C : \{0,1\}^k \rightarrow \{0,1\}$ of depth at most $d$, and $m \in \{0,1\}$ such*

*that* $(\mathbf{pk}, \mathbf{sk}_1, ..., \mathbf{sk}_N) \leftarrow \mathsf{TFHE.Setup}(1^\kappa, 1^d, \mathbb{A})$*,* $ct_i \leftarrow \mathsf{TFHE.Enc}(\mathbf{pk}, m)$*,* $\hat{ct} \leftarrow \mathsf{TFHE.Eval}(\mathbf{pk}, C, \mathbf{ct}_1, ..., \mathbf{ct}_k)$*, and* $\mathbf{p}_j \leftarrow \mathsf{TFHE.PartDec}(\hat{\mathbf{ct}}, \mathbf{sk}_j)$*, where* $i \in [k], j \in [N]$*. According to Definition 7, a TFHE is compact if there are* $|\hat{ct}| \leq poly(\kappa, d)$ *and* $|\mathbf{p}_i| \leq poly(\kappa, d, N)$ *for* $i \in [N]$*.*

**Definition 9** (Correctness)**.** *According to Definition 7, we say that a TFHE satisfies evaluation correctness if* $Pr[\mathsf{TFHE.FinDec}(\mathbf{pk}, \mathbf{B}) = C(m_1, ..., m_k)] = 1 - \varepsilon(\kappa)$*.*

### *D. Zero-Knowledge Argument of Knowledge*

The zero-knowledge argument of knowledge (ZKAoK) for a language $\mathcal{L}$ allows a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ that a series of instance $x$ is in $\mathcal{L}$ without revealing anything other than this statement. Further, a ZKAoK allows $\mathcal{P}$ to convince $\mathcal{V}$ with a witness $\omega$ evidencing a fact that $x$ is in $\mathcal{L}$, where $\mathcal{L}$ is defined by a relation predicate $\mathbf{P}_{\mathcal{L}(x,\omega)}$. Peikert and Shiehian [44] proposed a non-interactive zero-knowledge proof (NIZK) system for a wide class of NP $\mathcal{L}$ over DLWE. We employ the generic conversion proposed by Sahai [48]) to transform the NIZK [44] into a simulation-sound NIZK (SS-NIZK).

**Definition 10** (SS-NIZK)**.** *Let* $\mathcal{W}_\mathcal{L}(x)$ *denotes a generic set of witnesses. For any* $x \in \mathcal{L}$ *and* $\omega \in \mathcal{W}_\mathcal{L}(x)$*, there is* $\mathbf{P}_{\mathcal{L}(x,\omega)} = 1$*. An SS-NIZK includes the following three algorithms:*

- $\mathsf{pp} \leftarrow \mathsf{NIZK.Setup}(1^\kappa)$*: on input the security parameter* $\kappa$*, this algorithm outputs a common random string* crs*.*
- $\pi \leftarrow \mathsf{NIZK}.\mathcal{P}(crs, x, \omega)$*: on input* crs*,* $x \in \mathcal{L}$ *and* $\omega \in \mathcal{W}_\mathcal{L}(x)$*, this algorithm outputs a proof* $\pi \in \{0,1\}^{poly(\kappa)}$*.*
- $\{0,1\} \leftarrow \mathsf{NIZK}.\mathcal{V}(crs, x, \pi)$*. on input* crs*,* $x$ *and* $\pi$*, it outputs 1 if* $x \in \mathcal{L}$*. Otherwise, it outputs 0.*

Set the public parameters $pp = \{A, \mathbf{pk}_1, \hat{\mathbf{pk}}, H, \mathbf{C}_{\tilde{pw}}, \hat{\mathbf{C}}_{\tilde{pw}}, \mathbf{C}_{\mathbf{p}_j}\}$. We need to prove the language $\mathcal{L}_U^{st_0} = \{A, \mathbf{pk}_1, \hat{\mathbf{pk}}, H, \mathbf{C}_{\tilde{pw}}, \hat{\mathbf{C}}_{\tilde{pw}} | \exists(\mathbf{R}_{\tilde{pw}}, \tilde{pw})\ s.t.(\mathbf{C}_{\tilde{pw}}, \hat{\mathbf{C}}_{\tilde{pw}}) = (\mathsf{TFHE.Enc}(\mathbf{pk}_1, H(\tilde{pw})), \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, H(\tilde{pw})))\}$ and $\mathcal{L}_S^{st_0,i} = \{A, \mathbf{C}_{\mathbf{p}_j} | \exists(\mathbf{R}_i, \mathbf{r}_i, \mathbf{Com}_i, sk_i, \tilde{pk}, \lambda_i^z, \mathbf{p}_j)\ s.t. \mathbf{C}_{\mathbf{p}_j} = \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, \mathbf{p}_j)\}$. There the language $\mathcal{L}_U^{st_0}$ and $\mathcal{L}_S^{st_0,i}$ corresponding to the interaction information in our scheme

**Definition 11** (NIZK Security, [44] )**.** *The algorithm* $\Pi = (\mathsf{NIZK.Setup}, \mathsf{NIZK}.\mathcal{P}, \mathsf{NIZK}.\ \mathcal{V})$ *is a SS-NIZK for a relation* $R$*. The language* $\mathcal{L}$ *is defined by* $R$ *if the following holds.*

- ***Completeness.*** *Let* $x \in \mathcal{L}$*.* $\omega$ *is the witness. If* $\mathbf{P}_{\mathcal{L}(x,\omega)} = 1$*, the following probability is negligible in* $\kappa$*:*

$$1 - Pr\left[1 \leftarrow \mathsf{NIZK}.\mathcal{V}(crs, x, \pi) \middle| \begin{matrix} \mathsf{pp} \leftarrow \mathsf{NIZK.Setup}(1^\kappa) \\ \pi \leftarrow \mathsf{NIZK}.\mathcal{P}(crs, x, \omega) \end{matrix}\right].$$

- ***Soundness.*** *For any* $\mathcal{P}$*,* ***Ext*** *is a PPT extractor. For any input* $x$ *and* $\mathsf{pp} \leftarrow \mathsf{NIZK.Setup}(1^\kappa)$*, the following probability is* $\varepsilon(\kappa)$*:*

$$Pr[\langle \mathcal{P}(\omega), \mathcal{V} \rangle (\mathsf{crs}, x) = 1 \wedge (x, \omega) \notin \mathcal{L} | \omega \leftarrow \textbf{\textit{Ext}}^{\mathcal{P}}(\mathsf{crs}, x)].$$

- ***Zero knowledge.*** *There exists a PPT simulator* ***Sim*** *such that for any* $\mathcal{V}^*$ *with auxiliary information* ***aux****,* $(x, \omega) \in \mathcal{L}$*, it holds that:* $View(\langle \mathcal{P}(w), \mathcal{V}^* \rangle (\mathsf{crs}, x, \textbf{\textit{aux}})) \approx_c \textbf{\textit{Sim}}^{\mathcal{V}^*}(\mathsf{crs}, x, \textbf{\textit{aux}})$*. Where* $\textbf{\textit{Ext}}^{\mathcal{P}}$ *means that* ***Ext*** *has access to the entire process of* $\mathcal{P}$ *(including the randomness) and* $\textbf{\textit{Sim}}^{\mathcal{V}^*}$ *denotes that* ***Sim*** *captures the randomness from a polynomial-size space of* $\mathcal{V}^*$*.*

## V. Scheme Architecture and Security Model

**System Model.** We present a quantum-resistant password-protected data outsourcing for cloud storage, named QPause. Specifically, the user $U$ divides the data encryption key $sk$ into $N$ parts and stores them in $N$ key servers $S = \{S_1, ..., S_N\}$, respectively. The ciphertext of password-related information and data are stored in the cloud server $CS$. Similar to the architecture in Fig. 1, but with different specific operations. Notably, the main interactive operations are concentrated with $U$ and $S$ in QPause, which employs a basic password-protected secret sharing (PPSS) scheme as the subprotocol.

A password-protected secret sharing scheme allows the user to retrieve outsourcing data by interacting with at least $t+1$ responding to honest key servers with a correct human-memorizable password. We follow the definition of the first PPSS scheme proposed by Bagherzandi et al. [8]. Let $\mathbf{M}$ denote the message space, and $\mathcal{D}$ is the dictionary of passwords.

**Definition 12** (Password-protected secret sharing)**.** *A* $(t, N)$*-PPSS scheme for the message space* $\mathbf{M}$*, and the password dictionary* $\mathcal{D}$ *contain the following two polynomial algorithms:*

- $\mathsf{st} \leftarrow \mathsf{Init}(pw, m)$ *is a probabilistic algorithm executed by* $U$*.* $U$ *picks a secret key* $sk$*, and inputs a message* $m \in \mathbf{M}$ *and a password* $pw \in \mathcal{D}$*. The algorithm outputs* $st = \{st_0, st_1, ..., st_N\}$*, where* $st_0$ *is the outsourcing data, and* $\{st_1, ..., st_N\}$ *are shares of* $sk$ *among* $N$ *key servers.*
- $m' \leftarrow \mathsf{Rec}(\widetilde{pw}, st_0)$ *is an interactive protocol executed between the user* $U$ *and a subset of* $t+1$ *key servers indexed by an access structure* $\mathbb{S}$ *as follows:*
  - $m'/\perp \leftarrow \mathsf{RecU}(\widetilde{pw}, st_0)$*.* $U$ *inputs a password* $\widetilde{pw} \in \mathcal{D}$ *and* $st_0$ *to retrieve* $m'$ *by interacting with* $t+1$ *key servers in* $\mathbb{S}$*. The algorithm outputs* $m'$ *or* $\perp$*.*
  - $\mathsf{RetS}(st_i, st_0)$*. is executed by the key server* $S_i$ *in the* $\mathbb{S}$*.* $S_i$ *inputs the secret share* $st_i$ *and* $st_0$ *by interacting with* $U$*. There is no local output.*

**Definition 13** (Correctness)**.** *For any* $m \in \mathbf{M}$ *and* $pw \in \mathcal{D}$*,* $\mathsf{st} \leftarrow \mathsf{Init}(pw, m)$*. The probability* $Pr[m' = m] = 1$*, iff* $\mathsf{st} \leftarrow \mathsf{Init}(pw, m)$*,* $m' \leftarrow Ret(\widetilde{pw}, st_0)$ *and* $pw = \widetilde{pw}$*.*

**Security Model.** We employ the PPSS security model of Bagherzandi et al. [8] to characterize the security of our $(t, N)$-PPSS. At the high level, $\mathcal{A}$ can corrupt at most $t'$ servers to access the corresponding share $\{st_i\}$ and the outsourcing data $st_0$. $\mathcal{A}$ holds concurrent oracle access to Rec and RetS in Definition 12. The advantage of $\mathcal{A}$ is to distinguish between two PPSS instances initialized with two different messages $m_0, m_1 \in \mathbf{M}$, where $\mathbf{M}$ is a message space.

**Definition 14** (PPSS security)**.** *A* $(t, N)$*-PPSS scheme on dictionary* $\mathcal{D}$ *and message space* $\mathbf{M}$ *is* $(t, N, T, q_{user}, q_{send}, \varepsilon)$*-secure if for any* $m_0, m_1 \in \mathbf{M}$*, any set* $\hat{S}$ *with the size* $t' \leq t$*, and any PPT algorithm* $\mathcal{A}$ *with executing time* $T$*, there is*

$$|p^0 - p^1| \leq \left\lfloor \frac{q_{send}(\kappa)}{t - t' + 1} \right\rfloor \cdot \frac{1}{|\mathcal{D}|} + \varepsilon(\kappa)$$

*where* $p^b$ *is the probability that* $\mathcal{A}(m_0, m_1, st_0, st_{\hat{S}})$ *outputs 1 on access to* $q_{user}$ *sessions with* $\mathsf{RecU}(\widetilde{pw}, st_0)$*, and* $q_{send}$ *sessions with oracle* $\mathsf{RetS}(st_{\hat{S}}, st_0)$*, for* $\mathsf{st} \leftarrow \mathsf{Init}(pw, m_b)$*.*

Notably, $\mathcal{A}$ making at most $q_{send}$ online attacks, the adversary's advantage $Adv$ is denoted as $q_{send}(\kappa)/|\mathcal{D}| + \varepsilon(\kappa)$ for all dictionary sizes $|\mathcal{D}|$ in the existing uniform-model. Recent research [19], [14], [34] provided a rigorous analysis to constrain the adversary's advantage as $C' \cdot q_{send}^{s'}(\kappa) + \varepsilon(\kappa)$ for the Zipf parameters $C'$ and $s'$, with considering the password distribution follows the Zipf-distribution. We show that the advantages of the adversary are underestimated in the uniform-model in Section VIII. Further, we update the definition of the adversary's advantage in Definition 14 as follows:

$$|p^0 - p^1| \leq C' \cdot \left\lfloor \frac{q_{send}^{s'}(\kappa)}{t - t' + 1} \right\rfloor + \varepsilon(\kappa)$$

**Definition 15** (PPSS robustness [8])**.** *A PPSS scheme on dictionary $\mathcal{D}$ and message space $\mathbf{M}$ is $(T, \varepsilon)$-robust if for any $(m, pw) \in \mathbf{M} \times \mathcal{D}$, any $\widetilde{S}$ s.t. $n - |\widetilde{S}| \geq t+1$, and any PPT algorithm $\mathcal{A}$ with executing time $T$, the probability that $m' \neq m$, where $\mathsf{st} \leftarrow \mathsf{Init}(pw, m_b)$ and $m' \leftarrow \mathsf{RecU}(\widetilde{pw}, st_0)$ interacting with $\mathcal{A}(m, pw, st_{\widetilde{S}})$ and $\mathsf{RetS}(st_{\widetilde{S}}, st_0)$, is bounded by $\varepsilon$.*

**Definition 16** (PPSS soundness [8])**.** *A PPSS scheme on dictionary $\mathcal{D}$ and message space $\mathbf{M}$ is $(T, \varepsilon)$-sound if for any $(m, pw, \widetilde{pw}) \in \mathbf{M} \times \mathcal{D} \times \mathcal{D}$, and any PPT algorithm $\mathcal{A}$ with executing time $T$, the probability that $m' \notin \{s, \perp\}$, where $\mathsf{st} \leftarrow \mathsf{Init}(pw, m_b)$ and $m' \leftarrow \mathsf{RecU}(\widetilde{pw}, st_0)$ interacting with $\mathcal{A}(m, pw, \widetilde{pw}, st)$, is bounded by $\varepsilon$. We define weak soundness in the same way but restricting $\widetilde{pw}$ to $\widetilde{pw} = pw$.*

**Quantum resistance**. The most influential quantum attack algorithms are Shor's [30] and Grover's [41]. Quantum computers can efficiently solve large integer decomposition and discrete logarithm problems with Shor's algorithm. Grover algorithm allows quantum computers to speed up the search for unstructured databases and hash collisions. In order to avoid the effects caused by both algorithms above, we construct the PPSS scheme based on DLWE in definition 3, i.e., for any PPT $\mathcal{A}$, the advantage holds that $Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) \leq \varepsilon(\kappa)$.

Note that, the key reuse attack remains a threat to public key cryptography in the quantum setting [49] and includes two types of attacks: the signal leakage attack [16] and the key mismatch attack [17]. Specifically, 1) For signal leakage attacks, $\mathcal{A}$ impersonates a user with the malformed secret key to initiate a series of sessions, and then $\mathcal{A}$ observes changes in the return signals by the servers. 2) For key mismatch attacks, $\mathcal{A}$ as server sets special secret key share and error. Then, $\mathcal{A}$ recovers the secret key of the victim whose public key remains unchanged through multiple queries, and destroys the PPSS.

## VI. QPause: Our New Scheme

In this section, we present the quantum-resistant password-protected data outsourcing for cloud storage (QPause). First, we describe a basic lattice-based PPSS secure against honest but curious adversaries, assuming secure channels. Then, we extend the lattice-based PPSS to the QPause, which can address active threats and achieve security against malicious adversaries. To simplify the expression, we denote the steps of bit-by-bit encryption and circuit-solving ciphertext uniformly as encryption (and omit the expression for circuit depth).

### A. Password Protected Secret Sharing over Lattices

In this section, we present a basic password-protected secret sharing scheme over lattices. Our construction employs a fully homomorphic encryption protocol of Gentry et al. [38] with a threshold decryption from Shamir secret sharing [39], [43]. Fix the security parameter $\kappa$. Let $n, m, q$ and $\chi$ be an appropriately chosen DLWE parameters according to Definition 3, where $\mathcal{X}$ is a noise distribution, $n \in \mathbb{Z}^+$, $q = poly(n)$ prime power of 2, and $m = \Theta(nlog\ q)$. Let $\mathcal{N} = (n+1) \cdot (\lfloor log\ q \rfloor + 1)$, $\mathcal{D}$ denote the dictionary, and $M \in \mathbb{Z}_q$ is a standard message space. $H$ is a public collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_q$.

According to Definition 12, the basic PPSS scheme with secure channels consists of two algorithms as follows:

$\mathsf{Init}(pw, m)$: On input a password $pw \in \mathcal{D}$ and a secret message $m \in \mathbf{M}$, the algorithm samples a uniformly random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, a uniformly random vector $\mathbf{k}_\tau \leftarrow \mathbb{Z}_q^n$, where $\tau = \{1, 2\}$ an error vector $\mathbf{e}_\tau \leftarrow \mathcal{X}^m$. Let $pp = \{\mathbf{A}, \mathbf{k}_\tau, \mathbf{e}_\tau, c\}$, where $\tau = \{1, 2\}$ and $c$ is a constant. Let $d$ be the depth bound, and an access structure $\mathbb{S}$. Then, it sets $\mathbf{pk}, \mathbf{sk}_1, ..., \mathbf{sk}_N) \leftarrow \mathsf{TFHE.Setup}(1^\kappa, 1^d, \mathbb{S})$. Next, $\mathsf{Init}$ samples random matrix $\mathbf{R}_{pw}, \mathbf{R}_m \leftarrow \{0,1\}^{\mathcal{N} \times m}$, and encrypts $H(pw)$ and $m$: $\mathbf{C}_{pw} \leftarrow \mathsf{TFHE.Enc}(\mathbf{pk_1}, H(pw))$ and $\mathbf{C}_m \leftarrow \mathsf{TFHE.Enc}(\mathbf{pk_2}, m)$. Finally, the algorithm output $st = \{st_0, st_1, ..., st_N\}$, where the outsourcing data $st_0$ includes $\{\mathbf{pk}_1, H, \mathbf{C}_{pw}, \mathbf{C}_m\}$, and the secret sharing $st_i = \mathbf{sk}_i$ sends to the key server $S_i$ for $i \in [1, N]$.

$\mathsf{Rec}(\tilde{pw}, st_0)$: between the user $U$ inputs a password $\tilde{pw}$ and the outsourcing data $st_0$ downloaded from the cloud server, and the key server $\{S_j\}_{j=1}^n$ on inputs $st_j$ proceeds as follows:

1. $U$ samples a random matrix $\mathbf{R}_{\tilde{pw}} \leftarrow \{0,1\}^{\mathcal{N} \times m}$, and encrypts the $H(\tilde{pw})$ by computing $\mathbf{C}_{\tilde{pw}} \leftarrow \mathsf{TFHE.Enc}(\mathbf{pk}_1, H(\tilde{pw}))$. Then, $U$ sends $\mathbf{C}_{\tilde{pw}}$ to each $S_j$. Notably, there is $pw = \tilde{pw}$ for a legitimate user.
2. $S_j$ samples a random matrix $\mathbf{R}_j \leftarrow \{0,1\}^{m \times m}$, and randomizes the password, i.e., $S_j$ computes $\Delta_j = (\mathbf{C}_{\tilde{pw}} - \mathbf{C}_{pw}) \cdot \mathbf{R}_j$. Then, $S_j$ sends $\Delta_j$ to $U$.
3. $U$ picks a set $\mathbb{S}$ of $t+1$ servers (for the sake of clarity, we assume that $U$ picks the top $t+1$ servers.) and computes $\Delta = \sum_{j=1}^{t+1} \Delta_j$. For $j \in \mathbb{S}$, $U$ sends $(\mathbb{S}, \Delta)$ to each $S_j$.
4. $S_j$ executes $\mathbf{p}_j \leftarrow \mathsf{TFHE.PartDec}(\Delta + \mathbf{C}_m, \mathbf{sk}_j)$ and sends $\mathbf{p_j}$ to $U$. Specifically, $S_j$ computes $\lambda_j^z = (N!)^2 \cdot \prod_{i \in \mathbb{S}/\{j\}} \frac{-i}{j-i} \bmod q$ and executes the partial decryption, i.e., $S_j$ sends $\mathbf{p_j} = \lambda_j^z(\langle \mathbf{sk}_j, (\Delta + \mathbf{C}_m) \rangle + (N!)^2 \cdot \mathbf{e}_j)$ to $U$, where $\mathbf{e}_j \leftarrow \mathcal{X}^m$ and $\lambda_j^z \leq (N!)^3$.
5. $U$ computes $m_k^{bit} \leftarrow \mathsf{TFHE.FinDec}(\mathbf{sk}, \mathbf{B})$ and output $m$, where $m_k^{bit}$ is the $k$-th bit of $m$.

Notice that we modify the form of the public key, i.e., we added a multiplicative constant $c$ to the noise $\mathbf{e}$. This modification can maintain security while keeping the decryption noise as an integer multiple of $c$ [39]. In addition, at the fourth step of $\mathsf{Rec}(\tilde{pw}, st_0)$, we must add a noise $\mathbf{e}_j$ to prevent the adversary from performing Gaussian elimination to extract the information of the secret share $\mathbf{sk}_i$. However, the noise $\mathbf{e}_j$ is amplified by the Lagrange coefficient in the aggregation phase, which leads to decryption failure. We employ clear out their denominators [40] to limit the Lagrange coefficients to an integer. Specifically, Let the Lagrange coefficient $\lambda_i \in \mathbb{R}$.

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 be $\mathbb{Z}_p$ for a series of prime $p$ with $(N!)^3 \leq p$. Then, for every $1 \leq j \leq t$, the integer Lagrange coefficients $\lambda_j^z = (N!)^2 \cdot \prod_{i \neq j}^{t} \frac{-I_i}{(I_j - I_i)}$ is bounded $\lambda_j^z \leq (N!)^3$. Thus, the final form of adding noise to the partial decryption is $(N!)^2 \cdot \mathbf{e}_j$. We follow the instance of [Section 5.3.1, [39]], and set $\mathbf{e} \leq B$, $\mathbf{e}_j \leq B_{sm}$, where $B = \sigma\sqrt{n}$ and $B_{sm} = \frac{q - 4\sigma\sqrt{n}}{4(N!)^3 \cdot N}$.

**Lemma 1** (Correctness)**.** *Let $q, m, n, \sigma > 0$ depend on $\kappa$, and $\mathbf{e} \leq \sigma\sqrt{n}, \mathbf{e}_j \leq \frac{q - 4\sigma\sqrt{n}}{4(N!)^3 \cdot N}$. For any $m \in \mathbf{M}$ and $pw \in \mathcal{D}$, $\mathsf{st} \leftarrow \mathsf{Init}(pw, m)$. The user $U$ and each server $\{S_i\}_{i=1}^N$ execute $\mathsf{st} \leftarrow \mathsf{Init}(pw, m)$ and $m' \leftarrow \mathsf{Rec}(\tilde{pw}, st_0)$. The probability $Pr[m' = m] = 1$, iff $pw = \tilde{pw}$.*

*Proof:* When all the user $U$ and key servers $\{S_i\}_{i=1}^N$ are honest, at the fifth step of $\mathsf{Rec}(\tilde{pw}, st_0)$, there is $\mathbf{p} = \sum_{j=1}^{t+1} \mathbf{p_j} = \sum_{j=1}^{t+1} \lambda_j^z(\langle \mathbf{sk}_j, (\Delta + \mathbf{C}_m) \rangle + \mathbf{e}_j) = \langle \sum_{j=1}^{t+1} \lambda_j^z \cdot \mathbf{sk}_j, (\Delta + \mathbf{C}_m) \rangle + \sum_{j=1}^{t+1} \lambda_j^z \cdot \mathbf{e}_j = \langle \mathbf{sk}_2, ((\mathbf{C}_{\tilde{pw}} - \mathbf{C}_{pw}) \cdot \sum_{j=1}^{t+1} \mathbf{R}_j + \mathbf{C}_m) \rangle + \sum_{j=1}^{t+1} \lambda_j^z \cdot \mathbf{e}_j$, where $\mathbf{C}_{\tilde{pw}} - \mathbf{C}_{pw} = \mathsf{Flatten}((H(\tilde{pw}) - H(pw)) \cdot \mathbf{I}_{\mathcal{N}}) + \mathsf{BitDecomp}(\mathbf{R}_{\tilde{pw}} \cdot \hat{\mathbf{pk}}_{\mathbf{1}} - \mathbf{R}_{pw} \cdot \mathbf{pk}_1)$. Acorroding to Definition 6, the result of the decryption $\mathbf{p} = \frac{q}{2} \cdot (t+1) \cdot (H(\tilde{pw}) - H(pw)) + \frac{q}{2} \cdot m + c \cdot \mathbf{e} \cdot \hat{\mathbf{R}} + \sum_{j=1}^{t+1} \lambda_j^z \cdot \mathbf{e}_j$, where $\hat{\mathbf{R}} = \sum_{j=1}^{t+1} \mathbf{R}_j \cdot (\mathbf{R}_{\tilde{pw}} - \mathbf{R}_{pw}) + \mathbf{R}_m$ is a low-norm integer vector. The decryption noise is the inner product $c \cdot \mathbf{e} \cdot \hat{\mathbf{R}}$ is an integer multiple of $c$, and $\sum_{j=1}^{t+1} \lambda_j^z \cdot \mathbf{e}_j$ is an integer multiple of the integer Lagrange coefficients $\lambda_j^z = (N!)^2 \cdot \lambda_j$. Therefore, the message $m$ can be recovered with $\mathsf{TFHE.FinDec}$ iff $H(\tilde{pw}) - H(pw)$, i.e., $pw = \tilde{pw}$. □

**Theorem 1.** *According to the setting of our basic lattice-based PPSS scheme, let $\mathcal{A}$ can get $st_0$ from $CS$. $\mathcal{A}$ can access the client-side oracle $q_{user}$ times and the server-side oracle $q_{send}$ times. For any PPT $\mathcal{A}$, the advantage of obtaining data that*

$$Adv_{\mathcal{A}}^{\mathsf{PPSS}}(\kappa) \leq C' \cdot q_{send}^{s'}(\kappa) + Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) + \varepsilon(\kappa).$$

*Sketched proof.* The basic PPSS scheme with secure channels can ward off the eavesdropping attack. In addition, $\tilde{pw}$ is protected against offline dictionary guessing attacks by a fully homomorphic encryption (FHE) algorithm, i.e., the security follows from the hardness of $\mathrm{DLWE}_{n,q,\chi,m}$. Assuming both $U$ and $\{S_i\}_i^N$ firmly implement the PPSS. At the fourth step of the basic PPSS, $S_i$ executes the threshold decryption for $\Delta + \mathbf{C}_m$. Intuitively, $m$ is masked by the re-randomized password. Especially, if $\tilde{pw} \neq pw$, the variable $\frac{q}{2} \cdot (t+1) \cdot (H(\tilde{pw}) - H(pw))$ in $\mathbf{p}_i$ acts like a pseudorandom mask. This means that the server's response $\mathbf{p}_i$ is indistinguishable from the random value on the session while $(\tilde{pw} - pw) \neq 0$.

Next, we discuss the case of $\tilde{pw} = pw$, i.e., $\mathcal{A}$ guesses the correct password. Despite $m$ loses the mask of $\frac{q}{2} \cdot (t+1) \cdot (H(\tilde{pw}) - H(pw))$, the value $\mathbf{e} \cdot \hat{\mathbf{R}}$ still acts like a pseudorandom one-time pad masking the partial decryption, where $\hat{\mathbf{R}} = \sum_{j=1}^{t+1} \mathbf{R}_j \cdot (\mathbf{R}_{\tilde{pw}} - \mathbf{R}_{pw}) + \mathbf{R}_m$. Benefit from the Shamir secret sharing security assumption [43], when the adversary corrupts $t' \leq t$ key servers, there are also $t - t'$ masked values, which are indistinguishable from random values on the session. Consequently, the PPSS scheme holds the level of protection expected of password authentication [8]. The detailed security proof follows from the proof technique of the QPause with restrictions at Section VII.

### *B. QPause via PPSS*

We now propose QPause in Fig 3, a quantum resistant password protected data outsourcing for cloud storage. We first describe how to modify the basic PPSS to provide security against malicious adversaries without any secure channels. Then, we formally propose the construction of QPause. To meet the new challenges posed by changes in adversaries, we modify a series of steps of the basic PPSS as follows:

- We considers a specific session $sid = (S_1, ..., S_N, sid')$ and only accepts inputs from $S_i$ with the $sid$. $ID_j$ denotes server $ID$ of $j$-th Server session, $ID_j \in \{1, ..., N\}$.
- The optimization of the calculation sequence can further change the basic PPSS scheme to a round-optimal scheme. $U$ requests $st_0$ while determining the access structure $\mathbb{S}$ with $t+1$ servers, and sends the $\mathbb{S}$ together with $\mathbf{C}_{\tilde{pw}}$. Then, $S_j$ executes the partial decryption immediately after calculating the re-randomization password $\Delta_j$ at step 10 of RecS in Fig. 3. Finally, $U$ executes the final decryption to output the message $m$.
- In the step 3 of RecU, $\mathcal{A}$ can save $\mathbf{C}_{pw}$ to impersonate a legitimate $U$. Hence, $U$ needs to prove that the output $\mathbf{C}_{pw}$ uses a fresh random value $\mathbf{R}_{\tilde{pw}}$. We consider to encrypt $H(\tilde{pw})$ again, denoted as $\hat{\mathbf{C}}_{\tilde{pw}}$, with the same random value $\mathbf{R}_{\tilde{pw}}$ and a fresh public key $\hat{\mathbf{pk}}$, and then proving that the same $\mathbf{R}_{\tilde{pw}}$ is used for both $\mathbf{C}_{\tilde{pw}}$ and $\hat{\mathbf{C}}_{\tilde{pw}}$ by a lattice-based zero-knowledge in Section IV-D.
- When the basic PPSS scheme is without any secure channels, it is insecure that the response $\mathbf{p}_i$ of $S_i$ is sent back in cleartext. A passive eavesdropper on the channels between $U$ and $S$ can easily access $t+1$ shares and recover $m$. To counter such an eavesdropping attack, we employ the fully homomorphic encryption [38] again. In the first step of Rec, $U$ computes the public key $\tilde{\mathbf{pk}}$ with a random secret $\mathbf{r} \leftarrow \mathbb{Z}_q^n$. $U$ sends $\tilde{\mathbf{pk}}$ to each server in the access structure. $S_i$ encrypt $\mathbf{p}_i$ with $\tilde{\mathbf{pk}}$, denoted as $E(\mathbf{p}_i)$ and return it to $U$ instead of returning $\mathbf{p}_i$ directly in the fourth step of Rec. Finally, $U$ can decrypt $E(\mathbf{p}_i)$ with a staging $\mathbf{r}$ to get $\mathbf{p}_i$.
- We employ clear out their denominators [40] to limit the Lagrange coefficients to an integer. This makes the size of the ciphertext positively correlated with the number of servers $N$ violates the compactness. Inspired by Boneh et al. [39], we overcome this quandary by employing the universal thresholdizer to thresholdize the compact fully homomorphic encryption [38]. Furthermore, the instantiation of the universal thresholdizer can improve the robustness of the basic PPSS scheme, since the verification algorithm can detect the maliciously generated evaluation share. We need to amend the Init phase with the set of commitment parameters [50] to the secret key share $\{\mathbf{sk}_i\}_{i=1}^N$ of $S_i$. Concretely, Init samples an extra a uniformly random matrix $\overline{\mathbf{A}} \leftarrow \mathbb{Z}_q^{(n+1) \times (mn-m-n-1)}$, and a series of uniformly random vectors $\mathbf{r}_i \leftarrow \mathbb{Z}_q^{(m+n)}$. Let $\boldsymbol{A}_1 = [\boldsymbol{I}_n \ \boldsymbol{A}]$ and $\boldsymbol{A}_2 = [0^{(n+1) \times m} \ \boldsymbol{I}_{n+1} \ \overline{\mathbf{A}}]$.

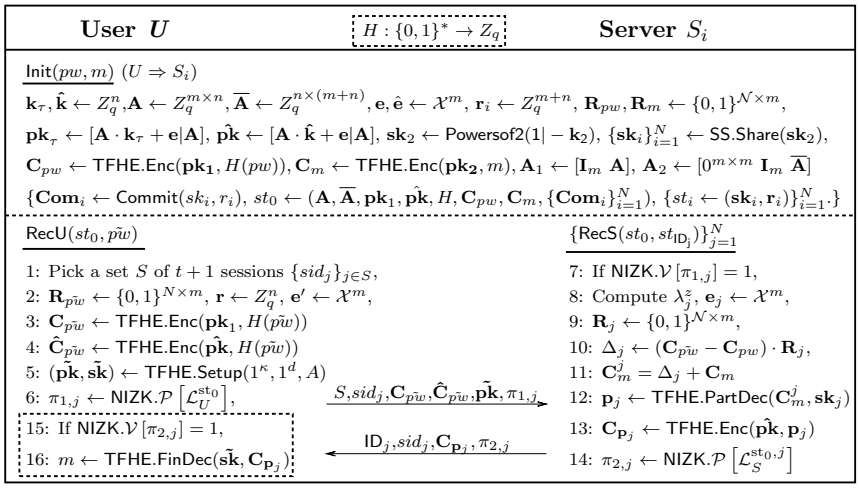


Fig. 3. Our QPause is secure against malicious adversaries with round-optimal, where $\tau = \{1, 2\}$ and the TFHE algorithm follows definition 7. The dashed box indicates that the user can recover data when receiving at least $t+1$ responses. The NIZK algorithm is the non-interactive zero-knowledge proof algorithm described in Section 10, where $\mathcal{L}_U^{st_0} = \{A, \mathbf{pk}_1, \hat{\mathbf{pk}}, H, \mathbf{C}_{p\tilde{w}}, \hat{\mathbf{C}}_{p\tilde{w}} | \exists(\mathbf{R}_{p\tilde{w}}, p\tilde{w})\ s.t. (\mathbf{C}_{p\tilde{w}}, \hat{\mathbf{C}}_{p\tilde{w}}) = (\mathsf{TFHE.Enc}(\mathbf{pk}_1, H(p\tilde{w})), \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, H(p\tilde{w})))\}$ and $\mathcal{L}_S^{st_0,i} = \{A, \mathbf{C}_{\mathbf{p}_j} | \exists(\mathbf{R}_i, \mathbf{r}_i, \mathbf{Com}_i, sk_i, \tilde{pk}, \lambda_i^z, \mathbf{p}_j)\ s.t. \mathbf{C}_{\mathbf{p}_j} = \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, \mathbf{p}_j)\}$.

Then, it outputs the commitment $\{\mathbf{Com}_i(\boldsymbol{sk}_i; \boldsymbol{r}) \leftarrow [\boldsymbol{A}_1 \cdot \boldsymbol{r}_i \| \boldsymbol{A}_2 \cdot \boldsymbol{r}_i + \boldsymbol{sk}_i]$, where $sk_i \in \mathbb{Z}_q^n$. We add a extra set $\{\boldsymbol{A}, \overline{\boldsymbol{A}}, \mathrm{Com}_i\}$ to $st_0$, and $\boldsymbol{r}_i$ to $\{st_i\}_{i=1}^N$.

According to Definition 10 and the analysis above, we need to prove that the two languages $\mathcal{L}_U^{st_0}, \mathcal{L}_S^{st_0,i}$ correspond to the messages transmitted by the two parties. All these messages are parameterized by common parameters $st_0 = (\boldsymbol{A}, \overline{\boldsymbol{A}}, \mathbf{pk}_1, \hat{\mathbf{pk}}, H, \mathbf{C}_{pw}, \mathbf{C}_m, \{\mathbf{Com}_i\}_{i=1}^N)$. $\mathcal{L}_U^{st_0} = \{A, \mathbf{pk}_1, \hat{\mathbf{pk}}, H, \mathbf{C}_{p\tilde{w}}, \hat{\mathbf{C}}_{p\tilde{w}} | \exists(\mathbf{R}_{p\tilde{w}}, p\tilde{w})\ \ s.t. (\mathbf{C}_{p\tilde{w}}, \hat{\mathbf{C}}_{p\tilde{w}}) = (\mathsf{TFHE.Enc}(\mathbf{pk}_1, H(p\tilde{w})), \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, H(p\tilde{w})))\}$ and $\mathcal{L}_S^{st_0,i} = \{A, \mathbf{C}_{\mathbf{p}_j} | \exists(\mathbf{R}_i, \mathbf{r}_i, \mathbf{Com}_i, sk_i, \tilde{pk}, \lambda_i^z, \mathbf{p}_j)\ s.t. \mathbf{C}_{\mathbf{p}_j} = \mathsf{TFHE.Enc}(\hat{\mathbf{pk}}, \mathbf{p}_j)\}$. The simulation-sound non-interactive zero-knowledge proofs (SS-NIZK) enable round-optimal and the messages well-formed. Peikert and Shiehian [44] propose an SS-NIZK based on Decision-$\mathsf{LWE}_{n,q,\chi,m}$, which can be transformed to SS-NIZK by the generic conversion of Sahai [48]. Thus, we can employ the SS-NIZK for $\mathcal{L}_U^{st_0}$ and $\mathcal{L}_S^{st_0,i}$.

### C. Further Discussion

**Multi-user devices.** The current setting of the QPause has the advantage of not requiring any device on the user side to store privacy information (the password can be recorded in the user's head). Users with weaker protection (as opposed to more protected servers) are not afraid of the data security risks associated with hacking their devices. Moreover, the QPause is easily extended to multi-user devices (e.g., laptops, iPads, mobile phones, etc.). Even if an adversary corrupts $t$ out of the $N$ servers, it cannot access the user's data stored in the cloud. However, in the round-optimal setting of our scheme, each server does not authenticate directly to the password input by a user. Although such a setting improves the privacy of user identities and the efficiency of communication, the system may be vulnerable to online password-guessing attacks [51].

Specifically, the server side does not know whether the user has completed the scheme, i.e. the server side does not see the status of the user side after the server returns a series of computation results. This provides a convenient condition for adversaries to launch online password-guessing attacks. The QPause cannot simply add an additional authentication to solve this problem above since if the server is corrupted, the adversary extracts the registered authentication information. A feasible method is that the server-side limits the number of user logins in a fixed period to prevent the adversary from performing online password-guessing attacks.

In addition, for a static set of servers, the adversary may be able to corrupt more than $t$ servers through a perpetual corruption attack [15]. We can employ the 0-share polynomial (The constant term of the secret sharing polynomial is 0) to update the server-side key share. This allows the adversary to corrupt the server for only a fixed period of time, thus preventing perpetual corruption attacks. However, there is no fundamental solution to the problem of the server being corrupted. Considering the desire of a certain part of users to have more control over their data [6], rather than leaving data security entirely in the hands of a subscribed cloud service.

Inspired by Bagherzandi et al. [8], we can improve security through user-controlled shares (Although this increases additional storage overhead and may result in a loss of robustness). Specifically, the initialization phase of the QPause generates $2N - t$ private key shares for the user. The user stores $N - t$ of these shares locally and shares the remaining shares with servers. Currently, the threshold of the scheme is $N$, i.e., at least $t$ servers provide private key shares $sk_i$ to participate in the computation when recovering the secret, and the user performs the calculation locally using $N - t$ shares. At this point, the adversary cannot extract the user's secret information through corruption attacks only. In addition, this setting can resist the online password guessing attack because the user side does not respond to adversary queries.

**Actively disrupt.** In practice, malicious adversaries can actively disrupt the system by launching denial-of-service attacks (DOS) or by tampering with the data. For DOS attacks, it is impossible to fully resist DOS attacks by password-based schemes. Service providers need to employ methods such as firewalls [52], [53] and intrusion detection systems [54], [55]. Notably, our QPause has the advantage of weakening DOS

attacks. Because our QPause is designed based on a distributed system with multiple servers, effectively improving the overall performance and capacity of the system. In particular, the reconstruction task can be completed when the $t+1$ servers in the $N$ servers are working. Our system can effectively distribute attack traffic, reduce the load on individual servers, and weaken the impact of DOS attacks.

## VII. Theoretical Analysis

In the following, we show that our scheme is provably secure in the formal model defined in Section V, under the Decision-$\mathsf{LWE}_{n,q,\chi,m}$, is intractable [37]. To the best of our knowledge, there is no known effective algorithm that can solve the Decision-$\mathsf{LWE}_{n,q,\chi,m}$ problem in the average case. In addition, we analyze the compactness, soundness, robustness (defined in Section V), and the complexity of our QPause.

### A. Security Analysis of QPause

We prove the security of our QPause based on subsection V with the PPSS security model, where the capabilities of the adversary $\mathcal{A}$ are modeled through queries and corrupt no more than $t$ servers in a fixed period. We employ the standard game-based proof to characterize the security of our QPause.

**Theorem 2** (Security)**.** *In QPause, a malicious $\mathcal{A}$ can get all public parameters, control the entire external network, and access client-side oracle and server-side oracle $q_u$ times and $q_s$ times, respectively. Furthermore, $\mathcal{A}$ can corrupt $t' \leq t$ servers $\{S_i\}$. For any PPT $\mathcal{A}$, the advantage of cracking QPause that:*

$$Adv_{\mathcal{A}}^{\mathsf{QPause}}(\kappa) \leq C' \cdot \left\lfloor \frac{q_{send}^{s'}(\kappa)}{t-t'+1} \right\rfloor + \frac{nq_s}{2^{m^2-1}} + (2q_u + 2q_s + 3) \cdot Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) + \varepsilon(\kappa).$$

*Proof:* Let $\mathcal{A}$ be an adversary against the semantic security of the QPause, running in time $T$. We describe a series of hybrid games $\mathbf{G}_n$ $(n = 1, ..., 8)$, all initialized on secret $m$, where $\mathbf{G}_0$ is the beginning to interact of $\mathcal{A}$ with QPause. With slight modifications explained below, the games ending in $\mathbf{G}_8$. Now the advantage of $\mathcal{A}$ is 0, and we can bound the difference between any adjacent games. For each game $\mathbf{G}_n$, we define the following events: 1) $\mathsf{Succ}_n$ denotes that $\mathcal{A}$ successfully guesses the bit $b$ involved in the query. 2) $\mathbf{G}_j^i$ denote the sub-games between $\mathbf{G}_{j-1}$ and $\mathbf{G}_j$, where $\mathbf{G}_{j-1}^i$ follows $\mathbf{G}_j$ calls on the first $i$ sessions. $k$ is the times of $\mathcal{A}$ access oracle.

**Game** $\mathbf{G}_0$. This game corresponds to the real attack. The adversary enters an encrypted password to at least $t-t'+1$ sessions executed by separate servers. According to the security assumptions and previous analysis, the advantage of $\mathcal{A}$ learning the secret is equivalent to guessing the correct password. By definition 14, we have: $Pr[succ_0] \leq C' \cdot \left\lfloor \frac{q_{send}^{s'}(\kappa)}{t-t'+1} \right\rfloor + \varepsilon(\kappa)$.

**Game** $\mathbf{G}_1$. In this game, $\mathcal{A}$ access a $RecU$ oracle by $q_u$ times and attempt to distinguish $\hat{\mathbf{C}}_{\widetilde{pw}} \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ and $\hat{\mathbf{C}}_{\tilde{pw}} \leftarrow \mathsf{Flatten}(H(\tilde{pw}) \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}_{\tilde{pw}} \cdot \hat{\mathbf{pk}}))$. $\mathcal{A}$ can get $st_0$ from $\mathsf{Init}$. The security of the fully homomorphic encryption relies on the public parameter matrix $\hat{\mathbf{pk}}$ being computationally indistinguishable from a uniformly random matrix in $\mathbb{Z}_q^{(n+1)\times m}$ by DLWE, even $e$ is expanded by a fixed constant multiple. The $(\mathbf{A} \cdot \hat{\mathbf{s}} + \hat{\mathbf{e}})$ and $c \cdot \mathbf{r}$ are computationally indistinguishable, and the scalar multiplication by a nonzero integer over a prime modulus is bijective. Thus, $\mathbf{G}_0^i - \mathbf{G}_0^{i-1} \leq Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$, and $Pr[succ_1] - Pr[succ_0] \leq q_u \cdot Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$.

**Game** $\mathbf{G}_2$. This game is similar to $\mathbf{G}_1$ except that $\mathcal{A}$ access a $RecS$ oracle by $q_s$ times to distinguish $\mathbf{p}_i \leftarrow \mathbb{Z}_q$ and $\{\mathbf{p}_i\} \leftarrow \mathbb{Z}_q$, else $\mathbf{p}_i \leftarrow \lambda_j^z \langle \mathbf{sk}_i, \Delta_j + \mathbf{C}_m \rangle + (N!)^2 \cdot \mathbf{e}_j$. Because of the presence of noise $(N!)^2 \cdot \mathbf{e}_j$, $\mathcal{A}$ cannot obtain information about $\Delta_j + \mathbf{C}_m$ by Gaussian elimination, unless A can solve the DLWE, i.e. $\mathbf{G}_1^i - \mathbf{G}_1^{i-1} \leq Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$, $i \in [1, q_s]$. Another way is by decrypting $\mathbf{C}_{\mathbf{p}_j} = \mathsf{Flatten}(\mathbf{p}_j \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}_j \cdot \tilde{\mathbf{pk}}))$. $\mathcal{A}$ access $RecU$ oracle by $q_u$ times and attempt to sample $\mathbf{r} \leftarrow \mathbb{Z}_q^n$. $\mathcal{A}$ samples $\mathbf{A} \leftarrow st_0$ and computes $\tilde{\mathbf{sk}} \leftarrow \mathsf{Powersof2}(1|-\mathbf{r})$. This means that $\mathcal{A}$ can distinguish $\mathbf{A} \cdot \mathbf{r} + \mathbf{e}'$ and $\mathbf{A} \cdot \mathbf{r}'$. By Definition 3, we have: $\mathbf{G}_1^j - \mathbf{G}_1^{j-1} \leq Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$, $j \in [1, q_u]$, and $Pr[succ_2] - Pr[succ_1] \leq (q_u + q_s) Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$.

**Game** $\mathbf{G}_3$. We define game $\mathbf{G}_3$ by $\mathcal{A}$ decrypted $\hat{\mathbf{C}}_{\tilde{pw}}$ to get $H(\tilde{pw})$. If $\mathcal{A}$ is successful, $\Delta_j = (\mathbf{C}_{\tilde{pw}} - \mathbf{C}_{pw}) \cdot \mathbf{R}_j$, else $\Delta_j \leftarrow \mathbb{Z}^{m \times m}$. For any PPT $\mathcal{A}$, the advantage of $H(\tilde{pw}) \leftarrow \mathsf{Dec}(\hat{\mathbf{C}}_{\tilde{pw}})$ without $\mathbf{sk}$ is $Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$. $\mathcal{A}$ can access a $RecS$ oracle $q_s$ times (Totally $n \cdot q_s$ times) to get $\Delta_j$. In $\mathbf{G}_2^i$ $(i \in [1, q_s])$, $\mathcal{A}$ guesses $R'_j \leftarrow \{0,1\}^{m \times m}$. We have $Adv_{\mathcal{A}}^{\mathbf{G}_2^i} \leq (\frac{1}{2})^{m^2}$, and $Pr[succ_3] - Pr[succ_2] \leq Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) + \frac{nq_s}{2^{m^2}}$.

**Game** $\mathbf{G}_4$. In this game, $\mathcal{A}$ access $\mathsf{Init}$ oracle to query $\mathbf{sk}_i$. For $\mathcal{A}$ can corrupt $t' \leq t$ servers. This situation is the same as that in $\mathbf{G}_3$. Because $\mathcal{A}$ does not know the correct password, the variable $\frac{q}{2} \cdot (t+1) \cdot (H(\widetilde{pw}) - H(pw))$ in $\mathbf{p}_i$ acts like a pseudorandom mask. $\mathcal{A}$ cannot distinguish whether the $\mathbf{sk}_i$ queried is a share of 0. Hence, $Pr[succ_4] - Pr[succ_3] = 0$.

**Game** $\mathbf{G}_5$. This game is similar to $\mathbf{G}_4$ except that $\mathcal{A}$ attempt to distinguish $\mathbf{C}_m \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{C}_m \leftarrow \mathsf{Flatten}(m \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}_m \cdot \mathbf{pk}))$. By Definition 3, we have $Pr[succ_5] - Pr[succ_4] = Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$.

**Game** $\mathbf{G}_6$. In this game, $\mathcal{A}$ access a $RecS$ oracle $q_s$ times (Totally $n \cdot q_s$ times) to get $\Delta_j$. In $\mathbf{G}_5^i$ $(i \in [1, q_s])$, $\mathcal{A}$ gusses $R'_j \leftarrow \{0,1\}^{m \times m}$. Thus, we have $Adv_{\mathcal{A}}^{\mathbf{G}_6^i} \leq (\frac{1}{2})^{m^2}$ and $Pr[succ_6] - Pr[succ_5] \leq \frac{nq_s}{2^{m^2}}$.

**Game** $\mathbf{G}_7$. This game is similar to $\mathbf{G}_6$ except that $\mathcal{A}$ attempt to distinguish $\mathbf{C}_{\widetilde{pw}} \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{C}_{\tilde{pw}} \leftarrow \mathsf{Flatten}(H(\tilde{pw}) \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}_{\tilde{pw}} \cdot \hat{\mathbf{pk}}))$. Clearly, $\mathbf{G}_6^j - \mathbf{G}_6^{j-1} \leq Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$, and $Pr[succ_7] - Pr[succ_6] = q_s \cdot Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$.

**Game** $\mathbf{G}_8$. Finally, in the $\mathbf{G}_8$, $\mathcal{A}$ access the $\mathsf{Init}$ oracle and attempt to distinguish $\mathbf{C}_{pw} \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{C}_{pw} \leftarrow \mathsf{Flatten}(H(pw) \cdot \mathbf{I}_{\mathcal{N}} + \mathsf{BitDecomp}(\mathbf{R}_{pw} \cdot \mathbf{pk}))$. By Definition 3 and Definition 6, we have:$Pr[succ_8] - Pr[succ_7] = Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa)$.

In summary, for any PPT $\mathcal{A}$, the advantage of disclosure of sabotaging QPause holds that: $Adv_{\mathcal{A}}^{\mathsf{QPause}}(\kappa) \leq C' \cdot \left\lfloor \frac{q_{send}^{s'}(\kappa)}{t-t'+1} \right\rfloor + \frac{nq_s}{2^{m^2-1}} + (2q_u + 2q_s + 3) \cdot Adv_{\mathcal{A}}^{\mathsf{DLWE}}(\kappa) + \varepsilon(\kappa)$.

We follow the Zipf model of Zhenai at Section VIII, where $|\mathcal{D}| = 5,260,229$, $C' = 0.0491866$ and $s' = 0.156027$. □

### B. Further Security Discussion

To the best of our knowledge, the prerequisite for the powerful arithmetic power of quantum computers is the existence of efficient problem solving quantum algorithms. The most influential quantum attack algorithms are Shor's [30] and Grover's algorithm [41]. Quantum computers can efficiently solve large integer decomposition and discrete logarithm problems with Shor's algorithm. The Grover algorithm allows quantum computers to threaten symmetric cryptographic algorithms and hash functions. Nevertheless, the Grover algorithm requires exponential levels of memory, so the symmetric encryption and hash functions can resist quantum attacks with appropriate parameter settings (e.g., double the security parameters).

From the security analysis in Theorem 2, it can be seen that our scheme can be reduced to Decision-$\mathsf{LWE}_{n,q,\chi,m}$ [37] to provide quantum resistance. Our security estimates are made by employing the "lwe-estimator" [56] to show the security of our QPause. In particular, we characterize LWE instances by parameters the ciphertext dimension $n$, the standard deviation $\sigma$, and the estimator defines noise rate $\alpha = \sigma \cdot \sqrt{2\pi}/q$. We follow the algorithm of determining security parameters [56], and determine two settings of parameters to make the QPause satisfy 82-bit and 128-bit quantum security, respectively. The parameters are selected as shown in Table I. To obtain more conservative parameters, we adopt the core-SVP methodology using the classical cost $2^{0.292*\beta}$ and quantum cost $2^{0.268*\beta}$.

Next, we discuss the compactness, soundness, and robustness of the QPause. According to the PPSS security model in Section V, we have the following three theorems.

**Theorem 3** (Compactness)**.** *Let $poly(\cdot)$ denote polynomial and $\mathbf{y}_i = \{\mathbf{C}_{\mathbf{p}_i}, \pi_{2,i}\}$. Our QPause is compact, i.e. there are $|\Delta_j + \mathbf{C}_m| \leq poly(\kappa)$ such that $|\mathbf{y}_i| \leq poly(\kappa, N)$ for $i \in [N]$.*

*Proof:* The compactness of our QPause is similar to the underlying TFHE scheme of Boneh et al. [39]. Fix the security parameter $\kappa$ and access structure $\mathbb{S}$. Let $(st_0, st_1, ..., st_N) \leftarrow \mathsf{Init}(pw, m)$, and $\{\mathbf{C}_{\mathbf{p}_i}, \pi_{2,i}\} \leftarrow \mathsf{RecS}$, where $i \in \mathbb{S}$. $\mathbf{C}_{\mathbf{p}_i} \in \mathbb{Z}_q^{(n+1)\times m}$ and there exists a polynomial $\mathbf{poly_1}(\cdot)$ s.t. $\mathbf{C}_{\mathbf{p}_i} \leq \mathbf{poly_1}(\kappa)$. In addition, $\pi_{2,i}$ is parameterized by

$$\mathcal{L}_S^{st_0,i} = \{A, \mathbf{C}_{\mathbf{p}_i} | \exists(\mathbf{R}_i, \mathbf{r}_i, \mathbf{Com}_i, sk_i, \widetilde{pk}, \lambda_i^z, \mathbf{p}_j)\}.$$

The $\mathbf{Com}_i = [\boldsymbol{A} \cdot \boldsymbol{r}_i | \overline{\boldsymbol{A}} \cdot \boldsymbol{r}_i + \boldsymbol{sk}_i]$. By the $|\mathbf{r}_i|, |\mathbf{sk}_i| \leq \mathsf{poly}(\kappa, N)$, there exists a polynomial $\mathsf{poly}_2(\cdot)$ s.t. $|\pi_{2,i}| \leq \mathsf{poly}(|\mathbf{C}_{\mathbf{p}_i}| + |\mathbf{Com}_i|) \leq \mathsf{poly}_2(\kappa, N)$. In summary, let $\mathbf{y}_i = \{\mathbf{C}_{\mathbf{p}_i}, \pi_{2,i}\}$ and $\mathsf{poly} = \mathsf{poly}_1 + \mathsf{poly}_2$, we have:

$$\mathbf{y}_i \leq \mathsf{poly}_1(\kappa) + \mathsf{poly}_2(\kappa, N) \leq \mathsf{poly}(\kappa, N). \quad \square$$

**Theorem 4** (Soundness)**.** *For any $(m, pw, \widetilde{pw}) \in \mathbf{M} \times \mathcal{D} \times \mathcal{D}$, and any PPT algorithm $\mathcal{A}$, the probability that $m' \notin \{s, \perp\}$, where $\mathsf{st} \leftarrow \mathsf{Init}(pw, m_b)$ and $m' \leftarrow \mathsf{RecU}(\widetilde{pw}, st_0)$ interacting with $\mathcal{A}(m, pw, \widetilde{pw}, st)$, is bounded by $\varepsilon$. Our QPause satisfies weak soundness by restricting $\widetilde{pw}$ to $\widetilde{pw} = pw$.*

*Proof:* The soundness of QPause follows from the soundness of SS-NIZK in Section IV-D. Because the SS-NIZK forces the server to perform the computation exactly according to QPause. We can add an additional public key $PK$ to $st_0$ and the corresponding secret key sign $m$. This way can extend QPause to strong soundness. The user needs to verify the validity of the signature by $PK$ before recovering $m$. $\square$

**Theorem 5** (Robustness)**.** *For any $(m, pw) \in \mathbf{M} \times \mathcal{D}$, any $\widetilde{S}$ s.t. $n - |\widetilde{S}| \geq t + 1$, and any PPT algorithm $\mathcal{A}$ with executing time $T$, the probability that $m' \neq m$, where $\mathsf{st} \leftarrow \mathsf{Init}(pw, m_b)$ and $m' \leftarrow \mathsf{RecU}(\widetilde{pw}, st_0)$ interacting with $\mathcal{A}(m, pw, st_{\widetilde{S}})$ and $\mathsf{RetS}(st_{\widetilde{S}}, st_0)$, is bounded by $\varepsilon$.*

*Proof:* For any PPT $\mathcal{A}$, if $\mathcal{A}$ outputs a fake partial evaluation $\mathbf{p}_i^*$ such that $\mathbf{p}_i^* \neq \lambda_j^z \langle \mathbf{sk}_i, \Delta_j + \mathbf{C}_m \rangle + (N!)^2 \cdot \mathbf{e}_j$ and $\mathcal{V}\left[\mathcal{L}_S^{\mathrm{st}_0,j}\right] = 1$. This means that $\mathbf{y}_i^* = (\mathbf{p}_i^*, \pi_{2,i}^*)$ satisfies: 1) $\mathcal{A}$ has a valid witness, or 2) $\mathcal{V}\left[\mathcal{L}_S^{\mathrm{st}_0,j}\right] = 1$ without a valid witness. For the first case, there is $\mathbf{p}_i^* \neq \mathbf{p}_i$. $\mathcal{A}$ has a valid witness iff $\mathcal{A}$ can find a randomness $r_i^*$ such that: $\mathbf{com}_i = \left[\boldsymbol{A}\ \overline{\boldsymbol{A}}\right]^T \cdot \boldsymbol{r}_i^* + [\mathbf{0}^n\ \boldsymbol{sk}_i^*]$. Nevertheless, this is a contradiction with the binding of commitment in [50]. For the second case, it clearly violated the soundness of the SS-NIZK in Section IV-D. Hence, our QPause is robust. $\square$

If the user is assumed to choose a weak and easily guessable password, no password-based cryptographic schemes will be secure. Thus, password-based cryptographic schemes (see recently standardized ones like [57] and [58]) generally implicitly assume a benign legitimate user that will follow the best password practices, such as no use of weak passwords. Note that, even if users choose a strong password, the entropy that passwords can provide is not high (i.e., 20-22 bits [23]). Thus, for security-critical applications, two-factor or multi-factor credentials are employed. In practice, there are a series of methods that can help users choose strong passwords, such as password strength meters [59] and password managers [60].

TABLE I
SECURITY LEVEL OF OUR SCHEME

| | n | q | $\sigma$ | $\beta$ | Classical | Quantum |
|---|---|---|---|---|---|---|
| PARMS 1 | 784 | $2^{32}-4$ | 56 | 307 | 90-bit | 82-bit |
| PARMS 2 | 1024 | $2^{32}-4$ | 128 | 479 | 140-bit | 128-bit |

TABLE II
RUNNING TIMES OF RELATED OPERATIONS (IN MS).

| Operations | $T_G$ | $T_E$ | $T_D$ | $T_S$ | $T_{exe}$ |
|---|---|---|---|---|---|
| Time | 0.354 | 0.412 | 0.326 | 0.403 | 0.854 |

### C. Complexity analysis

According to Section VI-B, our QPause includes sampling, matrix multiplication and addition, encryption and decryption, secret sharing, commitment, and zero-knowledge proof. We can achieve sampling with a complexity of $\mathcal{O}(1)$ based on the Alias method proposed by Walker and Alastair [61].

In the $\mathsf{Init}$ phase, the generation of public keys includes a matrix multiplication and a matrix addition. We employ the number theorem transform (NTT) [62] to speed up the matrix operations in LWE and the optimized complexity is $\mathcal{O}(nlog\ m)$. In addition, the fully homomorphic encryption in Definition 6 has a real complexity of $\mathcal{O}((nd)^\omega)$ [38], where $n$ is dimensional, $d$ is the depth of circuit $C$, and $\omega < 2.3727$. The complexity of the $\mathsf{SS.Share}$ is $\mathcal{O}(tN)$, which is related to the degree of the polynomial (i.e., threshold $t$) and the

number of servers $N$ [39]. Finally, the commitment [50] can also be adapted to NTT optimization, and the complexity is $\mathcal{O}(nlog\ m)$. In practice, the parameter settings meet $n \geq m \gg d > N \geq t \geq \omega$ [34], [38], [44]. Therefore, the computation complexity of the Init phase is $\mathcal{O}(n^{\omega})$.

The decryption of the fully homomorphic encryption in Definition 6 is essentially Regev decryption [38] with the complexity of $\mathcal{O}(log\ n)$ [37]. Furthermore, the complexity of the aggregation is squared with the threshold $t$. Hence, the complexity of TFHE.FinDec is $\mathcal{O}(t^2) + \mathcal{O}(log\ n)$. Finally, according to Peikert and Shiehian [44], the complexity of the prove algorithm is $\mathcal{O}(nlog\ m + log^2 m + 2m)$, and the complexity of the verify algorithm is $\mathcal{O}(nlog\ m + 2m)$. Thus, the computation complexity of the RecU and RecS are $\mathcal{O}(n^{\omega})$. From the above analysis, it can be seen that the main overhead of QPause is the fully homomorphic encryption operation. Therefore, our QPause can benefit from more efficient fully homomorphic encryption schemes, e.g., Heaws [63].

## VIII. Experiments

In this section, we evaluate the overheads and functions of our password-protected secret sharing (PPSS) schemes.

**Overheads.** We calculate the computation cost in terms of basic cryptographic operations. We denote a key pair generation as $T_G$. We write $T_E$ as the fully homomorphic encryption and $T_D$ as the decryption. $T_S$ denotes the $(t, N)$ Shamir secret sharing operation in Definition 4 and $T_{exe}$ denotes the exponentiation. Our implementation is in C++ language and complies with the NTL version 11.5.1, and the measurement is obtained on a workstation with an Intel(R) Core(TM) i7-8750H running at 2.20 GHz. The operating costs of basic cryptographic operations are shown in Table II.

To ensure a 128-bit quantum security level, we employ the recommended parameter set by NIST PQC round 2 [64]. Let $c$ is constant, we can obtain an LWE instance by parameters $n, q, \alpha, m$, where the dimension $m = n$, an odd prime $q \approx n^c$, and the noise rate $\alpha \approx n^{1/2-c}$. Specifically, we set $n = 1024$, $q = 2^{32} - 4 \approx 4,294,967,291$, $\sigma = \alpha \cdot q/\sqrt{2\pi} = 128$. The practical parameters for implementing our QPause can be found in the scripts of LWE-Frodo[1] and GSW-FHE[2].

We evaluate the computation, communication, and memory overheads of the traditional and quantum-resistant PKI-based schemes, including schemes of Bagherzandi et al. [8], Camenisch et al. [25], Jarecki et al. [28], Roy et al. [18], and our work, as shown in Table III. We adopt the threshold setting of Camenisch et al. [25], where $t = N = 2$. Our scheme performs $T_G$ twice in the initialization phase (one key pair is used to encrypt and decrypt the outsourced data, and another key pair is used to prove the freshness of the ciphertext in the recovery phase). On the one hand, the main overhead is homomorphic encryption operations. On the other hand, the initialization phase can be computed offline. Therefore, such additional overheads are acceptable in practice.

In addition, the additional generated key prevents the user from encrypting random numbers in the recovery phase. Concretely, we can reduce one operation of fully homomorphic encryption by generating additional keys in each phase, effectively reducing the computation overhead (22.5%), communication overhead (49.6%), and memory overhead (37%) in the recovery phase. Furthermore, Roy et al.[18] do not add noise to the output in the partial decryption phase, which may lead to the extraction of private key information by the adversary through Gaussian elimination. Our work solves this security problem by adding noise to the partial decryption and handling the noise to ensure successful decryption.

[1]https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf

[2]https://github.com/google/fully-homomorphic-encryption

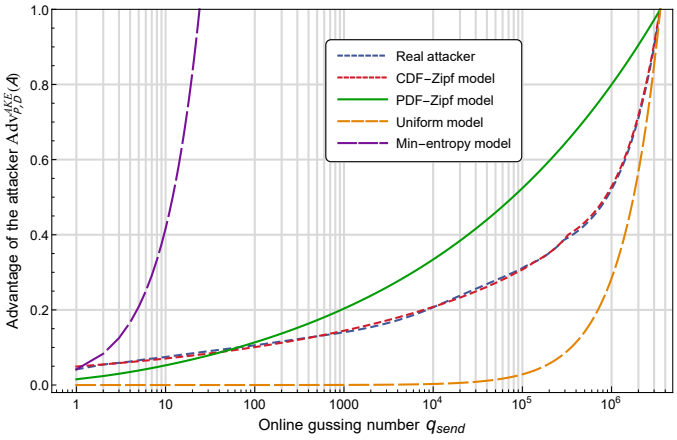


Fig. 4. Online guessing advantages of a real attacker, the CDF-Zipf modeled attacker, the PDF-Zipf modeled. The uniform-modeled attacker and min-entropy-modeled attacker (using the 5,260,229 passwords leaked from the dating site Zhenai). The overlap of the CDF-Zipf attacker with the real one indicates well prediction.

**Function.** A comparison between Bagherzandi et al. [8], Camenisch et al. [25], [45], [46], Jarecki et al. [28], [11], [29], [10], Yi et al. [47], Das et al.[5], Roy et al.[18] and our work is shown in Table IV. Compared to the $(N, N)$-threshold scheme of Das et al.[5], our QPause provides more options for users to recover data based on the assumption of secret sharing. On the one hand, users have more freedom to choose trusted servers to perform data recovery operations. On the other hand, users can employ an $(N, 2N - t)$ threshold scheme to have more control over their data, which is discussed at Section VI-C.

In terms of security models, the universally composable (UC) model appears to be more widely used [47], [11], [29], [10], [5]. However, in the case of a quantum attack, it is difficult to use the definition of UC security [29] since it is hardness to construct programmable random oracles in quantum random oracle model (ROM) [65]. To intuitively analyze the password protection of private information, we employ the ROM to portray the security of the scheme.

Next, we consider the impact of the different password distributions on security. From Fig. 4, it is clear that in the ROM, assuming that the password follows a uniform random distribution brings about a "relax" of the security reduction. Concretely, Fig. 4 shows that the advantages of the adversary are underestimated in the uniform model. Notably, the CDF-Zipf based formulation $C' \cdot q_{send}^{s'}(\kappa) + \varepsilon(\kappa)$ well approximates the real attacker's $Adv : q_{send} \in [1, |\mathcal{D}|]$ (Here we use the Zipf model of Zhenai, where $|\mathcal{D}| = 5,260,229$, $C' = 0.0491866$ and $s' = 0.156027$, the maximum deviation less than 0.491%).

This CDF-Zipf-based formulation is more accurate than other used formulations such as the Min-entropy model [66].

TABLE III
COMPARING THE COSTS OF TRADITIONAL [8], [25], [28] AND QUANTUM-RESISTANT PPSS SCHEME [18] WITH OUR WORK.

| | | Client | | Server | | |
|---|---|---|---|---|---|---|
| | Scheme | Comp.cost | Total time | Comp.cost | Total time | Comm.cost |
| Initialization | Bagherzandi et al. [8] * | $10T_{exe}$ | 8.51ms | - | - | $\mathcal{O}(N)$ |
| | Camenisch et al. [25] * | $18T_{exe}$ | 15.372ms | $11T_{exe}$ | 9.194ms | $\mathcal{O}(N)$ |
| | Jarecki et al. [28] * | $6T_{exe}$ | 5.124ms | - | - | $\mathcal{O}(N)$ |
| | Roy et al. [18] | $T_G + 2T_E + T_S$ | 1.581ms | - | - | $\mathcal{O}(N)$ |
| | Our work | $2T_G + 2T_E + T_S$ | 1.935ms | - | - | $\mathcal{O}(N)$ |
| Recovery | Bagherzandi et al. [8] * | $33T_{exe}$ | 28.182ms | $16T_{exe}$ | 13.664ms | $\mathcal{O}(t)$ |
| | Camenisch et al. [25] * | $19T_{exe}$ | 16.226ms | $26T_{exe}/30T_{exe}$ | 30.744ms/25.619ms | $\mathcal{O}(1)$ |
| | Jarecki et al. [28] * | $7T_{exe}$ | 5.978ms | $2T_{exe}$ | 1.68ms | $\mathcal{O}(t\ log\ N)$ |
| | Roy et al. [18] | $3T_E$ | 1.236ms | $T_E + T_D$ | 0.738ms | $\mathcal{O}(t)$ |
| | Our work | $T_G + 2T_E + T_D$ | 1.092ms | $T_E + T_D$ | 0.738ms | $\mathcal{O}(t)$ |

∗ Traditional (not post-quantum secure) PPSS schemes
† Comp.cost=Computation cost; Comm.cost=Communication cost.
‡ In the dual server PPSS scheme proposed by Camenisch et al. [25], the computational overhead of the two servers is different.

TABLE IV
COMPARISON AMONG RECENTLY PASSWORD-PROTECTED SHAMIR SHARING. CONSIST OF BAGHERZANDI ET AL. [8], CAMENISCH ET AL. [25], [45], [46], JARECKI ET AL. [10], [11], [28], [29], YI ET AL. [47], DAS ET AL.[5], ROY ET AL.[18], AND OUR WORK.

| | Threshold | Round | Password distribution | System model | Security model | Technology | ZK proof | Quantum security |
|---|---|---|---|---|---|---|---|---|
| Bagherzandi et al. (CCS'11) [8] | (t,N) | 2 | UR | PKI | ROM | HE | Y | N |
| Camenisch et al. (CCS'12) [25] | (2,2) | 1 | - | PKI | UC | HE | Y | N |
| Camenisch et al. (Crypto'14) [45] | (t,N) | 2 | - | PKI | UC | HE | Y | N |
| Camenisch et al. (PKC'15) [46] | (2,2) | 1 | - | PKI | UC | HE | Y | N |
| Jarecki et al. (ASIACRYPT'14) [28] | (t,N) | 1 | UR | CRS | ROM | OPRF | Y | N |
| Yi et al. (ESORICS'15) [47] | (t,N) | 1 | UR | CRS | UC | HE | N | N |
| Jarecki et al. (IEEE S&P'16) [11] | (t,N) | 1 | UR | CRS | UC | OPRF | N | N |
| Jarecki et al. (ACNS'17) [29] | (t,N) | 1 | UR | CRS | UC | OPRF | N | N |
| Jarecki et al. (CCS'19) [10] | (t,N) | 2 | UR | CRS | UC | OPRF | N | N |
| Das et al. (ASIACCS'20)[5] | (N,N) | 2 | UR | CRS | UC | OPRF | Y | N |
| Roy et al. (ACNS'21) [18] | (t,N) | 1 | UR | PKI | ROM | FHE | N | Y |
| Our work | (t,N) | 1 | Zipf | PKI | ROM | FHE | Y | Y |

† UR=Uniform random; "-"=Not to consider; Zipf=Zipf distribution; ZK=Zero-knowledge; CRS=Common reference string; Y =Yes; N=No.
‡ HE=Homomorphic encryption ; OPRF=Oblivious pseudorandom function; FHE=Fully homomorphic encryption;
∗ PKI=Public key infrastructure; UC=Universally Composable; ROM=random oracle model.

We use the accurate Zipf-based formulation for our QPause to achieve tighter security than PPSS schemes [8], [28], [18].

Compared to schemes without zero-knowledge proofs [28], [11], [29], [10], [47], [5], [18], [25], [45], [46], our solution can be better counter to malicious adversaries without secure channels. In addition, we implicitly build the universal thresholdizer [39] by incorporating commitment verification to make the scheme compact. A series of solutions [28], [47], [11], [10], [5] eliminate the public keys with the oblivious pseudorandom function (OPRF). However, the additional random numbers and noise introduced by password re-randomization over lattices may cause server-derived key recovery to fail (users may need to remember additional random numbers instead of just the password). We note that the verifiable OPRF scheme proposed by Albrecht et al. [67] may be used to construct PPSS protocols on the lattice without the PKI model, but this scheme cannot be used directly since low efficiency.

Finally, the security of our scheme can be reduced to the Decision-$\mathsf{LWE}_{n,q,\chi,m}$, in which the security of lattice-based hardness problem is reduced to the hardness of finding a relatively short vector over lattices. To the best of our knowledge, the Block-Korkin-Zolotarev (BKZ) algorithm [68] is the best solution to find the short vectors in the n-dimensional lattice. We can ensure that our QPause has 128-bit quantum security to resist the BKZ algorithm by setting suitable parameters.

## IX. LATTICE-BASED TPAKE FROM PPSS

We convert a threshold password-authenticated key exchange (TPAKE) protocol from our QPause. A TPAKE allows a user can complete authentication with a set of servers and securely establish session keys via a password in the public network. Bagherzandi et al. [8] show that a public key infrastructure (PKI)-based TPAKE protocol can be implemented by a chosen ciphertext attacks (CCA) secure encryption scheme $\Pi_E$, an existential unforgeability under adaptive chosen message attacks (EUF-CMA) signature scheme $\Pi_S$, and a secure password protected secret share (PPSS) scheme $\Pi_P$.

Roy et al. [18] demonstrated that the CCA secure lattice-based encryption scheme [69] and the EUF-CMA signature [70] are competent for the component TPAKE protocol. However, the existing lattice-based PPSS decomposes the secret agent at the server end, which will reveal the information about the secret key, as we discussed in Section III. Our

QPause eliminates obstacles and provides strong security PPSS components for TPAKE instantiation of quantum resistance.

At a high level, a user executes the key generation algorithms of $\Pi_E$ [69] and $\Pi_S$ [70] to get the encryption key pair $(\mathbf{pk}, \mathbf{sk})$ and the signing key pair $(\mathbf{ssk}_j, \mathbf{vk}_j)$ respectively. Then a user runs $\mathsf{Init}(pw, \mathbf{sk})$ of the QPause to complete the initialization ($\mathbf{sk}_i$ as secret information, $\mathbf{ssk}_j$ as sharing share). The user interacts with a set of servers based on Rec. In RecS, each server $S_j$ samples a key $\mathbf{k}_j \leftarrow \{0,1\}^\kappa$ and signs it by $\mathbf{ssk}_j$ to get $\mathsf{Sign}_j$. $S_j$ encrypts $(\mathbf{k}_j, Sign_j)$ with $\mathbf{pk}$ to get $\mathbf{C}_k$ and sends $\mathbf{C}_k$ to the user. The user recover $\mathbf{sk}$ by running RecU of QPause and decrypts $\mathbf{C}_k$ to obtain $(\mathbf{k}_j, Sign_j)$. If $Sign_j$ is valid, the user sets $\mathbf{k}_j$ as the session key with $S_j$.

We encapsulate the key with the CCA secure lattice-based encryption scheme [69] to ensure that adversaries can learn nothing about the session key. The EUF-CMA signature [70] guarantees that the transmitted information cannot be modified or rerouted. Hence, all $\mathbf{k}_j$ of users are independent of each other. Notably, $\mathbf{pk}$ is inevitable and may result in the key reuse attack. According to Section V, there are two attacks: signal leakage attacks [16] and key mismatch attacks [17].

For signal leakage attacks [16], $\mathcal{A}$ plays the role of the user. In our TPAKE, the user can recover the decryption key at any time using the password without caching it. The signal sent by the user does not contain $\mathbf{sk}$. Thus, our TPAKE is not influenced by signal leakage attacks. In addition, the decryption is performed locally and is protected by an implicit authentication (Section VI-B). Thus, unless an adversary has access to the password to execute a key mismatch attack [17].

## X. Conclusion

In this paper, our major goal is to construct a quantum-resistant password-protected data outsourcing scheme for cloud storage. It ensures that only a user who knows the correct password can retrieve data. To achieve this goal, we first construct a basic password-protected secret sharing (PPSS) scheme over lattices with a secure channel. Our PPSS enables users to retrieve data via their passwords secure against semi-honest adversaries.

Then, we formally propose a quantum-resistant password-protected data outsourcing scheme against malicious adversaries, named QPause. Our scheme requires no secure channels and achieves round-optimal. The security analysis shows that QPause can resist various attacks from quantum computing capable adversaries. In addition, we demonstrate the compactness, robustness, and soundness of the QPause. The comparison with related works shows that our QPause maintains the practicality of PPSS while guaranteeing quantum resistance.

## Acknowledgments

The authors thank the anonymous reviewers for their invaluable comments. This research was supported in part by the National Natural Science Foundation of China under Grant No. 62222208, by the Natural Science Foundation of Tianjin, China under Grants Nos. 21JCZDJC00190 and 21JCZXJC00100 and by Henan Key Laboratory of Network Cryptography Technology under Grant No. LNCT2022-A02.

## References

[1] D. Rydning, J. Reinsel, and J. Gantz, "The digitization of the world from edge to core," *Framingham: International Data Corporation 2018*.
[2] M. B. Hui, "Cloud services may become the biggest source of ddos attacks," 2019, https://www.freebuf.com/fevents/203988.html.
[3] M. Andrey, "Building a distributed network in the cloud: Using amazon ec2 to break passwords?" 2017, https://blog.elcomsoft.com/2017/08/breaking-passwords-in-the-cloud-using-amazon-p2-instances/.
[4] Y. Zhang, C. Xu, N. Cheng, and X. Shen., "Secure password-protected encryption key for deduplicated cloud storage systems," *IEEE Trans. Depend. Sec. Comput.*, vol. 19, no. 4, pp. 2789–2806, 2021.
[5] P. Das, J. Hesse, and A. Lehmann, "Dpase: Distributed password-authenticated symmetric encryption," in *Proc. ASIACCS 2022*.
[6] V. Mangipudi, U. Desai, M. Minaei, M. Mainack, and K. Aniket, "Uncovering impact of mental models towards adoption of multi-device crypto-wallets," 2022, https://eprint.iacr.org/2022/075.
[7] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Proc. FC 2010*, pp. 136–149.
[8] A. Bagherzandi, S. Jarecki, N. Saxena, and Y. Lu, "Password-protected secret sharing," in *Proc. ACM CCS 2011*, pp. 433–444.
[9] "Google cloud key management service," 2018, https://cloud.google.com/kms/.pdf.
[10] S. Jarecki, H. Krawczyk, and J. Resch, "Updatable oblivious key management for storage systems," in *Proc. ACM CCS 2019*.
[11] S. Jarecki, A. Kiayias, and H. Krawczyk, "Highly-efficient and composable password-protected secret sharing," in *Proc. IEEE EuroS&P 2016*.
[12] D. Boneh, K. Lewi, H. Montgomery, and A. Raghunathan, "Key homomorphic prfs and their applications," in *Proc. CRYPTO 2013*, 2013.
[13] M. Klooß, A. Lehmann, and A. Rupp, "(r) cca secure updatable encryption with integrity protection," in *Proc. EUROCRYPT 2019*.
[14] D. Wang and P. Wang, "Two birds with one stone: Two-factor authentication with security beyond conventional bound," *IEEE Trans. Depend.Sec. Comput.*, vol. 15, no. 4, pp. 708–722, 2018.
[15] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Proc. CRYPTO 1995*, pp. 339–352.
[16] J. Ding, S. Alsayigh, R. Saraswathy, S. Fluhrer, and X. Lin, "Leakage of signal function with reused keys in rlwe key exchange," in *Proc. ICC 2017*, pp. 1–6.
[17] J. Ding, S. Fluhrer, and S. Rv, "Complete attack on rlwe key exchange with reused keys, without signal leakage," in *Proc. ACISP 2018*, pp. 467–486.
[18] P. Roy, S. Dutta, W. Susilo, and R. Safavi-Naini, "Password protected secret sharing from lattices," in *Proc. ACNS 2021*, pp. 442–459.
[19] D. Wang, H. Cheng, P. Wang, , *et al.*, "Zipf's law in passwords," *IEEE Trans. Inf. Fore. Sec.*, vol. 12, no. 11, pp. 2776–2791, 2017.
[20] D. Wang, Y. Zou, Q. Dong, Y. Song, and X. Huang, "How to attack and generate honeywords," in *IEEE S&P 2022*, pp. 489–506.
[21] D. Wang, D. He, P. Wang, and C. Chu, "Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Trans. Depend.Sec. Comput.*, vol. 12, no. 4, pp. 428–442, 2015.
[22] J. Bonneau, C. Herley, P. Oorschot, and F. Stajano, "Passwords and the evolution of imperfect authentication," *ACM Commun.*, vol. 58, no. 7, pp. 78–87, 2015.
[23] H. Zhu, M. Xiao, D. Sherman, and M. Li, "Soundlock: A novel user authentication scheme for vr devices using auditory-pupillary response." in *Proc. NDSS 2023*, 2023, pp. 1–18.
[24] W. Burrand, D. Dodson, and W. Polk, "Electronic authentication guideline," *Nat. Inst. Standards Technol.*, 2013, doi: 10.6028/NIST.SP.XXX CODEN:NSPUE2.
[25] J. Camenisch, A. Lysyanskaya, and G. Neven, "Practical yet universally composable two-server password-authenticated secret sharing," in *Proc. ACM CCS 2012*, pp. 525–536.
[26] M. Abdalla, M. Cornejo, A. Nitulescu, and D. Pointcheval, "Robust password-protected secret sharing," in *Proc. ESORICS 2016*.
[27] J. Camenisch, A. Lehmann, and G. Neven, "Optimal distributed password verification," in *Proc. ACM CCS 2015*, pp. 182–194.
[28] 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*, pp. 233–253.
[29] S. Jarecki, A. Kiayias, H. Krawczyk, and J. Xu, "Toppss: cost-minimal password-protected secret sharing based on threshold oprf," in *Proc. ACNS 2017*, pp. 39–58.
[30] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proc. FOCS 1994*, pp. 124–134.

[31] T. Ladd, F. Jelezko, R. Laflamme, , *et al.*, "Quantum computers," *Nature*, vol. 464, no. 7285, pp. 45–53, 2010.

[32] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, pp. 79–99, 2018.

[33] G. Alagic *et al.*, "Status report on the first round of the nist post-quantum cryptography standardization process." 2019, https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf.

[34] Z. Li, D. Wang, and E. Morais, "Quantum-safe round-optimal password authentication for mobile devices," *IEEE Trans. Depend.Sec. Comput.*, vol. 19, no. 3, pp. 1885–1899, 2020.

[35] J. 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*, pp. 553–570.

[36] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. STOC 2008*.

[37] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, pp. 1–40, 2009.

[38] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Proc. CRYPTO 2013*, pp. 75–92.

[39] 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*, pp. 565–593.

[40] S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris, and H. Wee, "Functional encryption for threshold functions (or fuzzy IBE) from lattices," in *Proc. PKC 2012*, pp. 280–297.

[41] L. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. STOC 1996*, pp. 212–219.

[42] J. Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords," in *IEEE S& P 2012*, pp. 538–552.

[43] A. Shamir, "How to share a secret," *ACM Commun. 1979*, vol. 22, no. 11, pp. 612–613.

[44] C. Peikert and S. Shiehian, "Noninteractive zero knowledge for np from (plain) learning with errors," in *Proc. CRYPTO 2019*, pp. 89–114.

[45] J. Camenisch, A. Lehmann, A. Lysyanskaya, and G. Neven, "Memento: How to reconstruct your secrets from a single password in a hostile environment," in *Proc. CRYPTO 2014*, pp. 256–275.

[46] J. Camenisch, R. Enderlein, and G. Neven, "Two-server password-authenticated secret sharing uc-secure against transient corruptions," in *Proc. PKC 2015*, pp. 283–307.

[47] X. Yi, F. Hao, L. Chen, and J. K. Liu, "Practical threshold password-authenticated secret sharing protocol," in *Proc. ESORICS 2015*.

[48] A. Sahai, "Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security," in *Proc. FOCS 1999*, pp. 543–553.

[49] Q. Wang, D. Wang, C. Cheng, and D. He, "Quantum2fa: efficient quantum-resistant two-factor authentication scheme for mobile devices," *IEEE Trans. Depend.Sec. Comput.*, vol. 20, no. 1, pp. 193–208, 2021.

[50] C. Baum, I. Damgård, V. Lyubashevsky, S. Oechsner, and C. Peikert, "More efficient commitments from structured lattice assumptions," in *Proc. SCN 2018*, pp. 368–385.

[51] 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.*, vol. 20, no. 6, pp. 2297–2312, 2020.

[52] A. Liu and M. Gouda, "Diverse firewall design," *IEEE Trans. Parall. Distr.*, vol. 19, no. 9, pp. 1237–1251, 2008.

[53] C. Togay, A. Kasif, C. Catal, and B. Tekinerdogan, "A firewall policy anomaly detection framework for reliable network security," *IEEE Trans. Reliab.*, vol. 71, no. 1, pp. 339–347, 2021.

[54] G. Duan, H. Lv, H. Wang, and G. Feng, "Application of a dynamic line graph neural network for intrusion detection with semisupervised learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 699–714, 2022.

[55] D. Chou and M. Jiang, "A survey on data-driven network intrusion detection," *ACM Comp. Surv.*, vol. 54, no. 9, pp. 1–36, 2021.

[56] M. Albrecht, R. Player, and S. Scott, "On the concrete hardness of learning with errors," *J. Math. Crypto. 2015*, vol. 9, no. 3, pp. 169–203, 2015.

[57] D. Bourdrez, H. Krawczyk, K. Lewi, and C. Wood, "The opaque asymmetric pake protocol draft-irtf-cfrg-opaque-11." IETF Datatracker, June 8, 2023, https://datatracker.ietf.org/doc/draft-irtf-cfrg-opaque/.

[58] F. Hao, R. Metere, S. Shahandashti, and C. Dong., "Analyzing and patching speke in iso/iec," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 11, pp. 2844–2855, 2018.

[59] D. Wang, X. Shan, Q. Dong, Y. Shen, and C. Jia, "No single silver bullet: Measuring the accuracy of password strength meters," in *Proc. USENIX SEC 2023*, 2023, pp. 1–28.

[60] S. Zibaei, D. Malapaya, B. Mercier, A. Salehi, and J. Thorpe, "Do password managers nudge secure (random) passwords?" in *Proc. SOUPS 2022*, 2022, pp. 581–597.

[61] A. Walker and J. Alastair, "New fast method for generating discrete random numbers with arbitrary frequency distributions," *Electronics Letters*, vol. 8, no. 10, pp. 127–128, 1974.

[62] S. Hwajeong, K. Hyeokdong, K. Yongbeen, K. Kyungho, C. Seungju, K. Hyunjun, and J. Kyoungbae, "Fast number theoretic transform for ring-lwe on 8-bit avr embedded processor," *Sensors*, vol. 20, no. 7, pp. 20–39, 2020.

[63] F. Turan, S. Roy, and I. Verbauwhede, "Heaws: An accelerator for homomorphic encryption on the amazon aws fpga," *IEEE Transactions on Computers*, vol. 69, no. 8, pp. 1185–1196, 2020.

[64] G. Alagic *et al.*, "Status report on the second round of the nist post-quantum cryptography standardization process." 2020, https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf.

[65] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry, "Random oracles in a quantum world," in *Proc. ASI-ACRYPT 2011*, pp. 41–69.

[66] F. Benhamouda and D. Pointcheval, "Verifier-based password-authenticated key exchange: New models and constructions," 2013, https://eprint.iacr.org/2013/833.

[67] M. Albrecht, A. Davidson, A. Deo, and N. P. Smart, "Round-optimal verifiable oblivious pseudorandom functions from ideal lattices," in *Proc. PKC 2021*, pp. 261–289.

[68] C. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Math. program. 1994*, vol. 66, no. 1, pp. 181–199.

[69] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller." in *Proc. EuroCrypt 2012*, pp. 700–718.

[70] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," *J. of Crypto.*, vol. 25, pp. 601–639, 2012.


**Jingwei Jiang** is working toward a PhD degree from the College of Computer Science and Technology, Harbin Engineering University, China. As the first author, he has published papers at ESORICS 2022, the Chinese Journal of Computers, etc. His research interests include lattice-based cryptography, passwords, and authentication.

**Ding Wang** received his Ph.D. degree in Information Security at Peking University in 2017, and was supported by the "Boya Postdoctoral Fellowship" in Peking University from 2017 to 2019. Currently, he is a Full Professor at Nankai University. As the first author (or corresponding author), he has published more than 80 papers at venues like IEEE $S\&P$, ACM CCS, NDSS, Usenix Security, IEEE TDSC and IEEE TIFS. His research has been reported by over 200 media like Daily Mail, Forbes, IEEE Spectrum, and Communications of the ACM, appeared in the Elsevier 2017 "Article Selection Celebrating Computer Science Research in China", and resulted in the revision of the authentication guideline NIST SP800-63-2. He has been involved in the community as a PC Chair/TPC member for over 60 international conferences such as USENIX Security 2022/2020, NDSS 2024/2023, ACM CCS 2022/2021, PETS 2022-2024, ACSAC 2020-2023, RAID 2023, ACM AsiaCCS 2022/2021, IFIP SEC 2018-2021, ICICS 2018-2023, SPNCE 2020-2022. He has received the "ACM China Outstanding Doctoral Dissertation Award", the Best Paper Award at INSCRYPT 2018, the Outstanding Youth Award of China Association for Cryptologic Research, the Young Scientist Nomination Award for Powerful Nation, and the First Prize of Natural Science Award of Ministry of Education. His main research interests focus on passwords, authentication, and provable security.

**Guoyin Zhang** received his Ph.D. degree in the College of Computer Science and Technology from Harbin Engineering University Harbin, China, in 1999. Currently, he is a full professor at the College of Computer Science and Technology at Harbin Engineering University Harbin, China. His main research interests include network information security and embedded systems.