Zero-Knowledge Password Policy Check from Lattices

Khoa Nguyen, Benjamin Hong Meng Tan, Huaxiong Wang

Division of Mathematical Sciences,
School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore.
{khoantt,tanh0199,hxwang}@ntu.edu.sg

Abstract. Passwords are ubiquitous and most commonly used to authenticate users when logging into online services. Using high entropy passwords is critical to prevent unauthorized access and password policies emerged to enforce this requirement on passwords. However, with current methods of password storage, poor practices and server breaches have leaked many passwords to the public. To protect one's sensitive information in case of such events, passwords should be hidden from servers. Verifier-based password authenticated key exchange, proposed by Bellovin and Merrit (IEEE S&P, 1992), allows authenticated secure channels to be established with a hash of a password (verifier). Unfortunately, this restricts password policies as passwords cannot be checked from their verifier. To address this issue, Kiefer and Manulis (ESORICS 2014) proposed zero-knowledge password policy check (ZKPPC). A ZKPPC protocol allows users to prove in zero knowledge that a hash of the user's password satisfies the password policy required by the server. Unfortunately, their proposal is not quantum resistant with the use of discrete logarithm-based cryptographic tools and there are currently no other viable alternatives. In this work, we construct the first post-quantum ZKPPC using lattice-based tools. To this end, we introduce a new randomised password hashing scheme for ASCII-based passwords and design an accompanying zero-knowledge protocol for policy compliance. Interestingly, our proposal does not follow the framework established by Kiefer and Manulis and offers an alternate construction without homomorphic commitments. Although our protocol is not ready to be used in practice, we think it is an important first step towards a quantum-resistant privacy-preserving password-based authentication and key exchange system.

1 Introduction

One of the most common methods of user authentication is passwords when logging in to online services. So, it is very important that passwords in use have sufficient entropy and hard to guess for security. Password policies was introduced to guide users into choosing suitable passwords that are harder to guess. Ur et al. [51] discovered that users are more likely to choose easily guessable passwords in the absence of a password policy. Examining the password policies of over 70 web-sites, Florêncio and Herley [17] found that most require passwords with characters from at least one of four sets, digits, symbols, lowercase and uppercase letters and a minimum password length. Hence, it is reasonable to focus on password policies with a minimum password length, sets of valid characters and maybe constraints on the diversity of characters used.

Even with strong passwords and good policies, nothing can prevent leaks if servers do not properly store passwords. Improperly stored passwords can cause serious problems, as seen by hacks on LinkedIn [19] and Yahoo [47] and the web-site "Have I Been Pwned?" [25]. Sadly, such poor practices are not uncommon: many popular web-sites were discovered by Baumann et al. [4] to store password information in plaintext.

If servers cannot be trusted, then no password information should be stored there at all. Thus, protocols that do not require storing secret user information at external servers become necessary. However, even with secret passwords, password policies are important to enforce a base level of security against dictionary attacks, leaving a dilemma: how do users prove compliance of their password without revealing anything?

Kiefer and Manulis [30] showed how to address this problem with zero knowledge password policy check (ZKPPC). It enables blind registration: users register a password with a server and prove password policy conformance without revealing anything about their passwords, thereby solving the dilemma. With ZKPPC, some randomised password verification information is stored at the server and it does not leak information about the password, protecting against server compromises. Furthermore, ZKPPC allows a user to prove, without revealing any information, that the password conforms to the server's policy. Blind registration can be coupled with a verifier-based password-based authenticated key exchange (VPAKE) protocol to achieve a complete system for privacy-preserving password-based registration, authentication and key exchange. Password-based authenticated key exchange (PAKE) [6, 5, 20, 27, 14, 8] is a protocol that allows users to simultaneously authenticate themselves using passwords and perform key exchange. However, these protocols store passwords on the server and thus, users have to trust the security of the server's password storage and may be vulnerable to password leakages in the event of server compromise. Verifier-based PAKE [7, 5, 21, 11] extends PAKE to limit the damage caused by information leakage by storing a verifier instead. Verifiers are a means to check that users supplied the correct passwords and are usually hash values of passwords with a salt, which makes it hard to extract the passwords from verifiers.

A ZKPPC protocol allows users to prove that their password, committed in the verifier, satisfies some password policy. VPAKE can then be used to securely authenticate and establish keys whenever communication is required. Together, the password is never revealed, greatly increasing the user security over current standards. Passwords are harder to guess and no longer easily compromised by server breaches.

Kiefer and Manulis [30] proposed a generic construction of ZKPPC using homomorphic commitments and set membership proofs. In the same work, a concrete ZKPPC protocol was constructed using Pedersen commitments [45], whose security is based on the hardness of the discrete logarithm problem. As such, it is vulnerable to attacks from quantum adversaries due to Shor's algorithm [49] which solves the discrete logarithm problem in quantum polynomial time. With NIST issuing a call for proposals to standardize quantum resistant cryptography [44], it is clear that we need to prepare cryptographic schemes and protocols that are quantum resistant, in case a sufficiently powerful quantum computer is realized. As there is currently no proposal of ZKPPC protocol that has the potential to be quantum resistant, it is an interesting open problem to construct one.

OUR CONTRIBUTIONS AND TECHNIQUES. In this work, inspired by the attractiveness of ZKPPC protocols and the emergence of lattice-based cryptography as a strong quantum resistant candidate, we aim to construct a ZKPPC protocol from lattices. Our contribution is two-fold. We first design a randomised password hashing scheme based on the hardness of the Short Integer Solution (SIS) problem. We then construct a SIS-based statistical zero-knowledge argument of knowledge, which allows the client to convince the server that his secret password, committed in a given hash value, satisfies the server's policy. This yields the first ZKPPC protocol that still resists against quantum computers.

Our first technical challenge is to derive a password encoding mechanism that operates securely and interacts smoothly with available lattice-based cryptographic tools. In the discrete log setting considered in [30], passwords are mapped to large integers and then encoded as elements in a group of large order. Unfortunately, this does not translate well to the lattice setting as working with large-norm objects usually makes the construction less secure and less efficient. Therefore, a different method, which encodes passwords as small-norm objects, is desirable. To this end, we define a password encoding mechanism, that maps a password consisting of t characters to a binary vector of length $8t$, where each of the t blocks is the 8-bit representation of the ASCII value of the corresponding password character. To increase its entropy, we further shuffle the arrangement of those blocks using a random permutation, and then commit to the permuted vector as well as a binary encoding of the permutation via the SIS-based commitment scheme proposed by Kawachi, Tanaka and Xagawa [28]. This commitment value is then viewed as the randomised hash value of the password.

The next technical challenge is to prove in zero-knowledge that the committed password satisfies a policy of the form $f = \big((k_D, k_U, k_L, k_S), n_{\mathtt{min}}, n_{\mathtt{max}}\big)$, which demands that the password have length at least n_{min} and at most n_{max}, and contain at least k_D digits, k_S symbols, k_L lower-case and k_U upper-case letters. To this end, we will have to prove, for instance, that a committed length-8 block-vector belongs to the set of vectors encoding all 10 digits. We thus need a lattice-based sub-protocol for proving set membership. In the lattice-based world, a set membership argument system with logarithmic complexity in the cardinality of the set was proposed in [35], exploiting Stern-like protocols [50] and Merkle hash trees. However, the asymptotic efficiency does not come to the front when the underlying set has small, constant size. Here, we employ a different approach, which has linear complexity but is technically simpler and practically more efficient, based on the extend-then-permute technique for Stern's protocol, suggested by Ling et al. [37]. Finally, we use a general framework for Stern-like protocols, put forward by Libert et al. [33], to combine all of our sub-protocols for set membership and obtain a ZKPPC protocol.

From a practical point of view, our lattice-based ZKPPC protocol is not yet ready to be used: for a typical setting of parameters, an execution with soundness error 2^{-30} has communication cost around 900 KB. We, however, believe that there are much room for improvement and view this work as the first step in designing post-quantum privacy-preserving password-based authentication and key exchange systems.

RELATED WORK. The only construction of ZKPPC was proposed by Kiefer and Manulis [30] using Pedersen commitments [45] and a randomised password hashing scheme introduced in the same work. It commits each character individually and uses set membership proofs to prove compliance of the entire password to a password policy. The password hash is the sum of the committed characters and thus is linked to the set membership proofs through the homomorphic property of the commitments used. As mentioned previously, their protocol is vulnerable to quantum adversaries and greater diversity is desirable.

To improve the efficiency of secure password registration for VPAKE [31] and two server PAKE [32], Kiefer and Manulis proposed blind password registration (BPR), a new class of cryptographic protocols that prevent password leakage from the server. Using techinques introduced in [30], Kiefer and Manulis used an efficient shuffling proof from [18] to achieve $\mathcal{O}(1)$ number of set membership proofs instead of $\mathcal{O}(n_{max})$ in ZKPPC. However, the security model considered for BPR is only suitable for honest but curious participants. The security of ZKPPC is defined to prevent malicious users from registering bad passwords that do not conform to the given password policy. Malicious servers also do not gain any information on the registered password from running the ZKPPC protocol. Overall, the security model of BPR is weaker than the capabilities of ZKPPC and available instantiations are not resistant to quantum adversaries.

An alternate approach using symmetric key primitives, secure set-based policy checking (SPC), to check password policy was proposed in [16]. Policies are represented set-theoretically as monotone access structures and are mapped to linear secret sharing schemes (LSSS). Then, checking policy compliance corresponds to verifying whether some set is in the access structure, i.e. if the set of shares can reconstruct the secret in the LSSS. To obtain a privacy-preserving protocol for SPC, the oblivious bloom intersection (OBI) from [15] is used. The server constructs an LSSS that only users who fulfil the policy can obtain the right shares from the OBI and recover the secret. Knowledge of the secret is proved with a hash of the secret with the transcript of the protocol execution and identities of the two parties, tying the protocol to the proof of knowledge. In the proposed SPC protocol, the one-more-RSA assumption is used to guarantee that the password registration protocol is sound when used by a malicious client. Thus, in the presence of a quantum adversary, the SPC protocol cannot be considered sound anymore. Since the focus is on quantum resistant blind registration of passwords with malicious participants, the SPC protocol is insufficient.

ZERO-KNOWLEDGE PROOFS IN LATTICE-BASED CRYPTOGRAPHY. Early work on interactive and non-interactive proof systems [23, 43, 46] for lattices exploited the geometric structure of worst-case lattice problems, and are not generally applicable in lattice-based cryptography. More recent methods of proving relations appearing in lattice-based cryptosystems belong to the following two main families.

The first family, introduced by Lyubashevsky [40, 41], uses "rejection sampling" techniques, and lead to relatively efficient proofs of knowledge of small secret vectors [9, 2, 12, 13], and proofs of linear and multiplicative relations among committed values [10, 3] in the ideal lattice setting. However, due to the nature of "rejection sampling", there is a tiny probability that even an honest prover may fail to convince the verifier: i.e., protocols in this family do not have perfect completeness. Furthermore, when proving knowledge of vectors with norm bound β, the knowledge extractor of these protocols is only guaranteed to produce witnesses of norm bound $g \cdot \beta$, for some factor $g > 1$. This factor, called "soundness slack" in [2, 12], may be undesirable: if an extracted witness has to be used in the security proof to solve a challenge SIS instance, we need the $\mathsf{SIS}_{g\cdot\beta}$ assumption, which is stronger than the SIS_{β} assumption required by the protocol itself. Moreover, in some sophisticated cryptographic constructions such as the zero-knowledge password policy check protocol considered in this work, the coordinates of extracted vectors are expected to be in $\{0, 1\}$ and/or satisfy a specific pattern. Such issues seem hard to tackle using this family of protocols.

The second family, initiated by Ling *et al.* [37], use "decomposition-extension" techniques in lattice-based analogues [28] of Stern's protocol [50]. These are less efficient than those of the first family because each protocol execution admits a constant soundness error, and require repeating protocols $\omega(\log n)$ times, for a security parameter n, to achieve negligible soundness error. On the upside, Stern-like protocols have perfect completeness and can handle a wide range of lattice-based relations [38, 35, 36, 33, 34, 39], especially when witnesses have to not only be small or binary, but also certain prescribed arrangement of coordinates. Furthermore, unlike protocols of the first family, the extractor of Stern-like protocols can output witness vectors with the same properties expected of valid witnesses. This feature is often crucial in the design of advanced protocols involving ZK proofs. In addition, the "soundness slack" issue is completely avoided, so the hardness assumptions are kept "in place".

ORGANIZATION. In the next section, we define notations used in the paper and briefly describe the building blocks for our ZKPPC protocol. Following that, in Section 3, we instantiate the building blocks and ZKPPC protocol with lattices-based primitives. Finally, we summarize and conclude in Section 4.

2 Preliminaries

NOTATION. We assume all vectors are column vectors. A vector $\mathbf{x}$ with coordinates $x_1, \ldots, x_m$ is written as $\mathbf{x} = (x_1, \ldots, x_m)$. For simplicity, concatenation of $\mathbf{x} \in \mathbb{R}^k$ and $\mathbf{y} \in \mathbb{R}^m$ is denoted with $(\mathbf{x}\|\mathbf{y}) \in \mathbb{R}^{k+m}$. Column-wise concatenation of matrices $\mathbf{A} \in \mathbb{R}^{n\times k}$ and $\mathbf{B} \in \mathbb{R}^{n\times m}$ is denoted by $[\mathbf{A} \,|\, \mathbf{B}] \in \mathbb{R}^{n\times(k+m)}$. If S is a finite set, then $x \xleftarrow{\$} S$ means that x is chosen uniformly at random over S. For a positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$ and $\mathsf{negl}(n)$ denotes a negligible function in n. The set of all permutations of n elements is denoted by $\mathcal{S}_n$. All logarithms are of base 2.

2.1 Some Lattice-Based Cryptographic Ingredients

We first recall the average-case problem SIS and its link to worst-case lattice problems.

Definition 1 **($\mathsf{SIS}^{\infty}_{n,m,q,\beta}$ [1, 22]).** *Given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, find a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$ such that $\|\mathbf{x}\|_\infty \leq \beta$ and $\mathbf{A} \cdot \mathbf{x} = \mathbf{0} \bmod q$.*

The hardness of the SIS is guaranteed by the worst-case to average-case reduction from lattice problems. If $m, \beta = \mathsf{poly}(n)$, and $q > \beta \cdot \widetilde{\mathcal{O}}(\sqrt{n})$, then the $\mathsf{SIS}^{\infty}_{n,m,q,\beta}$ problem is at least as hard as the worst-case lattice problem SIVP_γ for some $\gamma = \beta \cdot \widetilde{\mathcal{O}}(\sqrt{nm})$ (see, e.g., [22, 42]).

The KTX commitment scheme. In this work, we employ the SIS-based commitment scheme proposed by Kawachi, Tanaka and Xagawa [28] (KTX). The scheme, with two flavours, works with lattice parameter n, prime modulus $q = \widetilde{\mathcal{O}}(n)$, and dimension $m = 2n\lceil \log q \rceil$.

In the variant that commits t bits, for some fixed $t = \mathsf{poly}(n)$, the commitment key is $(\mathbf{A}, \mathbf{B}) \xleftarrow{\$} \mathbb{Z}_q^{n \times t} \times \mathbb{Z}_q^{n \times m}$. To commit $\mathbf{x} \in \{0,1\}^t$, one samples randomness $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$, and outputs the commitment $\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r} \bmod q$. Then, to open $\mathbf{c}$, one reveals $\mathbf{x} \in \{0,1\}^t$ and $\mathbf{r} \in \{0,1\}^m$.

If there exists two valid openings $(\mathbf{x}_1, \mathbf{r}_1)$ and $(\mathbf{x}_2, \mathbf{r}_2)$ for the same commitment $\mathbf{c}$ and $\mathbf{x}_1 \neq \mathbf{x}_2$, then one can compute a solution to the $\mathsf{SIS}^{\infty}_{n,m+t,q,1}$ problem associated with the uniformly random matrix $[\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n \times (m+t)}$. On the other hand, by the left-over hash lemma [48], the distribution of a valid commitment $\mathbf{c}$ is statistically close to uniform over $\mathbb{Z}_q^n$ which implies that it is statistically hiding.

Kawachi et al. [28] extended the above t-bit commitment scheme to a string commitment scheme $\mathsf{COM} : \{0,1\}^* \times \{0,1\}^m \to \mathbb{Z}_q^n$. The extended scheme shares the same characteristics, statistically hiding from the parameters set and computationally binding under the SIS assumption.

In this work, we use the former variant to commit to passwords, and use COM as a building block for Stern-like zero-knowledge protocols.

2.2 Zero-Knowledge Argument Systems and Stern-like Protocols

We work with statistical zero-knowledge argument systems, interactive protocols where the zero-knowledge property holds against *any* cheating verifier and the soundness property holds against *computationally bounded* cheating provers. More formally, let the set of statements-witnesses $\mathrm{R} = \{(y, w)\} \in \{0,1\}^* \times \{0,1\}^*$ be an NP relation. A two-party game $\langle \mathcal{P}, \mathcal{V} \rangle$ is called an interactive argument system for the relation R with soundness error e if two conditions hold:

- $\mathsf{Completeness}$. If $(y, w) \in \mathrm{R}$ then $\Pr\big[\langle \mathcal{P}(y, w), \mathcal{V}(y) \rangle = 1\big] = 1$.
- $\mathsf{Soundness}$. If $(y, w) \notin \mathrm{R}$, then $\forall$ PPT $\widehat{\mathcal{P}}$: $\Pr[\langle \widehat{\mathcal{P}}(y, w), \mathcal{V}(y) \rangle = 1] \leq e$.

Here and henceforth, PPT denotes probabilistic polynomial time. An argument system is statistical zero-knowledge if for any $\widehat{\mathcal{V}}(y)$, there exists a PPT simulator $\mathcal{S}(y)$ which produces a simulated transcript that is statistically close to that of the real interaction between $\mathcal{P}(y, w)$ and $\widehat{\mathcal{V}}(y)$. A related notion is argument of knowledge, which requires the witness-extended emulation property. For 3 move protocols (*i.e.*, commitment-challenge-response), witness-extended emulation is implied by *special soundness* [24], which assumes the existence of a PPT extractor, taking as input a set of valid transcripts with respect to all possible values of the "challenge" to the same "commitment", and returning w' such that $(y, w') \in \mathrm{R}$.

Stern-like protocols. The statistical zero-knowledge arguments of knowledge presented in this work are Stern-like [50] protocols. In particular, they are Σ-protocols as defined in [26, 9], where 3 valid transcripts are needed for extraction instead of just 2. Stern's protocol was originally proposed for code-based cryptography, and adapted to lattices by Kawachi et al. [28]. It was subsequently empowered by Ling et al. [37] to handle the matrix-vector relations associated with the SIS and inhomogeneous SIS problems and extended to design several lattice-based schemes: group signatures [38, 35, 36, 33, 39], and group encryption [34].

The basic protocol has 3 moves. With COM, the KTX string commitment scheme [28], we get a statistical zero-knowledge argument of knowledge (ZKAoK) with perfect completeness, constant soundness error $2/3$, and communication cost $\mathcal{O}(|w| \cdot \log q)$, where $|w|$ is the total bit-size of the secret vectors.

An abstraction of Stern's protocol. We recall an abstraction of Stern's protocol, proposed in [33]. Let n, ℓ, q be positive integers, where $\ell \geq n$, $q \geq 2$, and VALID be a subset of $\{0,1\}^{\ell}$. Suppose $\mathcal{S}$ is a finite set and every $\phi \in \mathcal{S}$ is associated with a permutation Γ_{ϕ} of ℓ elements, satisfying the following conditions:

$$\begin{cases} \mathbf{w} \in \mathsf{VALID} \Leftrightarrow \mathit{\Gamma}_\phi(\mathbf{w}) \in \mathsf{VALID}, \\ \text{If } \mathbf{w} \in \mathsf{VALID} \text{ and } \phi \text{ is uniform in } \mathcal{S}, \text{ then } \mathit{\Gamma}_\phi(\mathbf{w}) \text{ is uniform in } \mathsf{VALID}. \end{cases} \tag{1}$$

We aim to construct a statistical ZKAoK for the following abstract relation:

$$\mathrm{R}_{\text{abstract}} = \big\{(\mathbf{M}, \mathbf{v}), \mathbf{w} \in \mathbb{Z}_q^{n\times\ell} \times \mathbb{Z}_q^n \times \mathsf{VALID} : \mathbf{M}\cdot\mathbf{w} = \mathbf{v} \bmod q.\big\}$$

Stern's original protocol has $\mathsf{VALID} = \{\mathbf{w} \in \{0,1\}^\ell : \mathsf{wt}(\mathbf{w}) = k\}$, where $\mathsf{wt}(\cdot)$ denotes the Hamming weight and $k < \ell$ for some given k, $\mathcal{S} = \mathcal{S}_\ell$ – the symmetric group on ℓ elements, and $\mathit{\Gamma}_\phi(\mathbf{w}) = \phi(\mathbf{w})$.

The conditions in (1) are key to prove that $\mathbf{w} \in \mathsf{VALID}$ in ZK: The prover $\mathcal{P}$ samples $\phi \xleftarrow{\$} \mathcal{S}$ and the verifier $\mathcal{V}$ checks that $\mathit{\Gamma}_\phi(\mathbf{w}) \in \mathsf{VALID}$; no additional information about $\mathbf{w}$ is revealed to $\mathcal{V}$ due to the randomness of ϕ. Furthermore, to prove in ZK that $\mathbf{M}\cdot\mathbf{w} = \mathbf{v} \bmod q$ holds, $\mathcal{P}$ samples $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^\ell$ to mask $\mathbf{w}$, and convinces $\mathcal{V}$ instead that $\mathbf{M}\cdot(\mathbf{w}+\mathbf{r}_w) = \mathbf{M}\cdot\mathbf{r}_w + \mathbf{v} \bmod q$.

We describe the interaction between $\mathcal{P}$ and $\mathcal{V}$ in Figure 1. A statistically hiding and computationally binding string commitment scheme COM, e.g. the scheme in Section 2.1, is used.

1. **Commitment:** Prover $\mathcal{P}$ samples $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^\ell$, $\phi \xleftarrow{\$} \mathcal{S}$ and randomness ρ_1, ρ_2, ρ_3 for COM.
   Then, a commitment $\mathrm{CMT} = \big(C_1, C_2, C_3\big)$ is sent to the verifier $\mathcal{V}$, where
   $$C_1 = \mathsf{COM}(\phi, \mathbf{M}\cdot\mathbf{r}_w \bmod q; \rho_1), \quad C_2 = \mathsf{COM}(\mathit{\Gamma}_\phi(\mathbf{r}_w); \rho_2), \quad C_3 = \mathsf{COM}(\mathit{\Gamma}_\phi(\mathbf{w}+\mathbf{r}_w \bmod q); \rho_3).$$
2. **Challenge:** $\mathcal{V}$ sends a challenge $Ch \xleftarrow{\$} \{1,2,3\}$ to $\mathcal{P}$.
3. **Response:** Based on Ch, $\mathcal{P}$ sends RSP computed as follows:
   - $Ch = 1$: Let $\mathbf{t}_w = \mathit{\Gamma}_\phi(\mathbf{w})$, $\mathbf{t}_r = \mathit{\Gamma}_\phi(\mathbf{r}_w)$, and $\mathrm{RSP} = (\mathbf{t}_w, \mathbf{t}_r, \rho_2, \rho_3)$.
   - $Ch = 2$: Let $\phi_2 = \phi$, $\mathbf{w}_2 = \mathbf{w} + \mathbf{r}_w \bmod q$, and $\mathrm{RSP} = (\phi_2, \mathbf{w}_2, \rho_1, \rho_3)$.
   - $Ch = 3$: Let $\phi_3 = \phi$, $\mathbf{w}_3 = \mathbf{r}_w$, and $\mathrm{RSP} = (\phi_3, \mathbf{w}_3, \rho_1, \rho_2)$.

Verification: Receiving RSP, $\mathcal{V}$ proceeds as follows:

- $Ch = 1$: Check that $\mathbf{t}_w \in \mathsf{VALID}$, $C_2 = \mathsf{COM}(\mathbf{t}_r; \rho_2)$, $C_3 = \mathsf{COM}(\mathbf{t}_w + \mathbf{t}_r \bmod q; \rho_3)$.
- $Ch = 2$: Check that $C_1 = \mathsf{COM}(\phi_2, \mathbf{M}\cdot\mathbf{w}_2 - \mathbf{v} \bmod q; \rho_1)$, $C_3 = \mathsf{COM}(\mathit{\Gamma}_{\phi_2}(\mathbf{w}_2); \rho_3)$.
- $Ch = 3$: Check that $C_1 = \mathsf{COM}(\phi_3, \mathbf{M}\cdot\mathbf{w}_3; \rho_1)$, $C_2 = \mathsf{COM}(\mathit{\Gamma}_{\phi_3}(\mathbf{w}_3); \rho_2)$.

In each case, $\mathcal{V}$ outputs 1 if and only if all the conditions hold.

Fig. 1: Stern-like ZKAoK for the relation $\mathrm{R}_{\text{abstract}}$.

The properties of the protocol are summarized in Theorem 1.

Theorem 1 ([33]). *Assuming that* COM *is a statistically hiding and computationally binding string commitment scheme, the protocol in Figure 1 is a statistical* ZKAoK *with perfect completeness, soundness error* $2/3$*, and communication cost* $\mathcal{O}(\ell \log q)$*. In particular:*

- *There exists a polynomial-time simulator that, on input* $(\mathbf{M}, \mathbf{v})$*, outputs an accepted transcript statistically close to that produced by the real prover.*
- *There exists a polynomial-time knowledge extractor that, on input a commitment* CMT *and* 3 *valid responses* $(\mathrm{RSP}_1, \mathrm{RSP}_2, \mathrm{RSP}_3)$ *to all* 3 *possible values of the challenge* Ch*, outputs* $\mathbf{w}' \in \mathsf{VALID}$ *such that* $\mathbf{M}\cdot\mathbf{w}' = \mathbf{v} \bmod q$.

The proof of the Theorem 1, which appeared in [33], employs standard simulation and extraction techniques for Stern-like protocols [28, 37]. The proof is provided in Appendix A for the sake of completeness.

2.3 Password Strings and Password Policies

Next, we present the models of password strings and policies, adapted from [30].

Password Strings. We consider password strings pw over the set of 94 printable characters Σ_{all} in the ASCII alphabet Σ_{ASCII}, where $\Sigma_{\mathsf{all}} = \Sigma_D \cup \Sigma_S \cup \Sigma_L \cup \Sigma_U \subset \Sigma_{\mathsf{ASCII}}$ is split into four disjoint subsets:

- The set of 10 digits $\Sigma_D = \{0, 1, \ldots, 9\}$;
- The set of 32 symbols $\Sigma_S = \{$!”#\$%&’()*+,-./ :;<=>?@ [\]^_ ‘{|} ~ $\}$;
- The set of 26 lower case letters, $\Sigma_L = \{a, b, \ldots, z\}$;
- The set of 26 upper case letters, $\Sigma_U = \{A, B, \ldots, Z\}$.

We denote by `Dict` a general dictionary containing all strings that can be formed from the characters in Σ_{all}. A password string $pw = (c_1, c_2, \ldots, c_k) \in \Sigma_{\mathsf{all}}^k \subset \mathtt{Dict}$ of length k is an ordered multi-set of characters $c_1, \ldots, c_k \in \Sigma_{\mathsf{all}}$.

Password Policies. A password policy $f = \big((k_D, k_S, k_L, k_U), n_{\mathsf{min}}, n_{\mathsf{max}}\big)$ has six components, a minimum length n_{min}, maximum length n_{max}, and integers k_D, k_S, k_L and k_U that indicate the minimum number of digits, symbols, upper-case and lower-case letters, respectively, a password string must contain. We say that $f(pw) = \mathtt{true}$ if and only if policy f is satisfied by the password string pw. For instance,

1. Policy $f = \big((1, 1, 1, 1), 8, 16\big)$ indicates that password strings must be between 8 and 16 characters and contain at least one digit, one symbol, one lower-case and one upper-case letters.
2. Policy $f = \big((0, 2, 0, 1), 10, 14\big)$ demands that password strings must be between 10 and 14 characters, including at least two symbols and one upper-case letter.

Remark 1. In practice, password policies typically do not specify n_{max} but we can simply fix a number that upper-bounds all reasonable password lengths.

2.4 Randomised Password Hashing and Zero-Knowledge Password Policy Check

We now recall the notions of randomised password hashing and zero-knowledge password policy check. Our presentation follows [30, 29].

Randomised Password Hashing. This mechanism aims to compute some password verification information that can be used later in more advanced protocols (e.g., ZKPPC and VPAKE). In order to prevent off-line dictionary attacks, the computation process is randomised via a pre-hash salt and hash salt. More formally, a randomised password hashing scheme $\mathcal{H}$ is a tuple of 5 algorithms $\mathcal{H} = (\mathsf{Setup}, \mathsf{PreSalt}, \mathsf{PreHash}, \mathsf{Salt}, \mathsf{Hash})$, defined as follows.

- $\mathsf{Setup}(\lambda)$: On input security parameter λ, generate public parameters pp, including the descriptions of the salt spaces $\mathbb{S}_P$ and $\mathbb{S}_H$.
- $\mathsf{PreSalt}(pp)$: On input pp, output a random pre-hash salt $s_P \in \mathbb{S}_P$.
- $\mathsf{PreHash}(pp, pw, s_P)$: On input pp, password pw and pre-hash salt s_P, output a pre-hash value P.
- $\mathsf{Salt}(pp)$: On input pp, output a random hash salt $s_H \in \mathbb{S}_H$.
- $\mathsf{Hash}(pp, P, s_P, s_H)$: On input pp, pre-hash value P, pre-hash salt s_P and hash salt s_H, output a hash value $\mathbf{h}$.

A secure randomised password hashing scheme $\mathcal{H}$ must satisfy 5 requirements: *pre-image resistance*, *second pre-image resistance*, *pre-hash entropy preservation*, *entropy preservation* and *password hiding*.

- Pre-image resistance (or tight one-wayness in [11]): Let $pp \leftarrow \mathsf{Setup}(\lambda)$ and $\texttt{Dict}$ be a dictionary of min-entropy β. $Hash(\cdot)$ is a function such that $(H_i, s_{H_i}) \leftarrow Hash(\cdot)$, where $s_{H_i} \leftarrow \mathsf{Salt}(pp)$ and $H_i \leftarrow \mathsf{Hash}(pp, P_i, s_{P_i}, s_{H_i})$. $P_i \leftarrow \mathsf{PreHash}(pp, pw_i, s_{P_i})$ with $s_{P_i} \leftarrow \mathsf{PreSalt}(pp)$ and $pw_i \overset{\$}{\leftarrow} \texttt{Dict}$. P_i is stored by $Hash(\cdot)$ and there is a function $\texttt{Verify}(i, P)$ such that $\texttt{Verify}(i, P) = 1$ if $P = P_i$.
  For all PPT adversaries $\mathcal{A}$ running in time at most t, there exists a negligible function $\varepsilon(\cdot)$ such that
$$Pr[(i, P) \leftarrow \mathcal{A}^{Hash(\cdot)});\ \texttt{Verify}(i, P) = 1] \leq \frac{\alpha t}{2^{\beta} t_{\mathsf{PreHash}}} + \varepsilon(\lambda)$$
  for small α and t_{PreHash}, the running time of $\mathsf{PreHash}$.
- Second pre-image resistance: For all PPT adversaries $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that for $P' \leftarrow \mathcal{A}(pp, P, s_H)$,
$$Pr\big[\big(P' \neq P\big) \wedge \big(\mathsf{Hash}(pp, P, s_H) = \mathsf{Hash}(pp, P', s_H)\big)\big] \leq \varepsilon(\lambda),$$
  where $pp \leftarrow \mathsf{Setup}(\lambda)$, $s_P \leftarrow \mathsf{PreSalt}(pp)$, $s_H \leftarrow \mathsf{Salt}(pp)$ and $P \leftarrow \mathsf{Hash}(pp, pw, s_P)$ for any $pw \in \texttt{Dict}$.
- Pre-hash entropy preservation: For all dictionaries $\texttt{Dict}$ that are samplable in polynomial time with min-entropy β and any PPT adversary $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that for $(P, s_P) \leftarrow \mathcal{A}(pp)$ with $pp \leftarrow \mathsf{Setup}(\lambda)$ and random password $pw \overset{\$}{\leftarrow} \texttt{Dict}$,
$$Pr\big[\big(s_P \in \mathbb{S}_P\big) \wedge \big(P = \mathsf{PreHash}(pp, pw, s_P)\big)\big] \leq 2^{-\beta} + \varepsilon(\lambda).$$
- Entropy preservation: For all min-entropy β polynomial-time samplable dictionaries $\texttt{Dict}$ and any PPT adversary $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that for $(H, s_P, s_H) \leftarrow \mathcal{A}(pp)$
$$Pr\big[\big(s_P \in \mathbb{S}_P\big) \wedge \big(s_H \in \mathbb{S}_H \wedge H = \mathsf{Hash}(pp, pw, s_P, s_H)\big)\big] \leq 2^{-\beta} + \varepsilon(\lambda),$$
  where $pp \leftarrow \mathsf{Setup}(\lambda)$ and $pw \overset{\$}{\leftarrow} \texttt{Dict}$.
- Password hiding: For all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1(pp)$ outputs two equal length passwords pw_0, pw_1 for $pp \leftarrow \mathsf{Setup}(\lambda)$ and $\mathcal{A}_2(H)$ outputs a bit b' for $H \leftarrow \mathsf{Hash}(pp, P, s_P, s_H)$, where $s_H \leftarrow \mathsf{Salt}(\lambda)$, $s_P \leftarrow \mathsf{PreSalt}(\lambda)$ and $P \leftarrow \mathsf{PreHash}(pp, pw_b, s_P)$ for a random bit $b \overset{\$}{\leftarrow} \{0, 1\}$, there exists a negligible function $\varepsilon(\cdot)$ such that
$$\big|Pr[b = b'] - \tfrac{1}{2}\big| \leq \varepsilon(\lambda).$$

Zero-Knowledge Password Policy Check. Let $\mathcal{H} = (\mathsf{Setup}, \mathsf{PreSalt}, \mathsf{PreHash}, \mathsf{Salt}, \mathsf{Hash})$ be a randomised password hashing scheme. A password policy check (PPC) is an interactive protocol between a client and server where the password policy $f = \big((k_D, k_S, k_L, k_U), n_{\mathsf{min}}, n_{\mathsf{max}}\big)$ of the server and public parameters $pp \leftarrow \mathsf{Setup}(\lambda)$ are used as common inputs. At the end of the execution, the server accepts a hash value $\mathbf{h}$ of any password pw of the client's choice if and only if $f(pw) = \texttt{true}$. A PPC protocol is an argument of knowledge of the password pw and ssrandomness $s_P \leftarrow \mathsf{PreSalt}(pp)$, $s_H \leftarrow \mathsf{Salt}(pp)$ used for hashing. To prevent leaking the password to the server, one additionally requires that the protocol be zero-knowledge.

More formally, a zero-knowledge PPC protocol is an interactive protocol between a prover (client) and verifier (server), in which, given $(pp, f, \mathbf{h})$ the former convinces the later in zero-knowledge that the former knows pw and randomness (s_P, s_H) such that:
$$f(pw) = \texttt{true} \qquad \text{and} \qquad \mathsf{Hash}(pp, P, s_P, s_H) = \mathbf{h},$$
where $P \leftarrow \mathsf{PreHash}(pp, pw, s_P)$.

3 Our Constructions

To construct randomised password hashing schemes and ZKPPC protocols from concrete computational assumptions, the first challenge is to derive a password encoding mechanism that operates securely and interacts smoothly with the hashing and zero-knowledge layers. In the discrete log setting considered in [30], passwords are mapped to large integers and then encoded as elements in a group of large order. Unfortunately, this does not translate well to the lattice setting as working with large-norm objects usually reduces the security and efficiency of the construction. Therefore, a different method, which encodes passwords as small-norm objects, is desirable. In this work, we will therefore use binary vectors.

Let $\mathsf{bin}(\cdot)$ be the function that maps non-negative integers to their binary decomposition. For any character c encoded in ASCII, let $\texttt{ASCII}(c) \in [0, 255]$ be its code. Then, we define $\mathsf{enc}(c)$ for an ASCII encoded character c and $\mathsf{enc}(pw)$ for some length-t password $pw = (c_1, \ldots, c_t) \in \Sigma^t$ as

$$\mathsf{enc}(c) = \mathsf{bin}(\texttt{ASCII}(c)) \in \{0,1\}^8,$$
$$\mathsf{encode}(pw) = \big(\mathsf{enc}(c_1)\|\ldots\|\mathsf{enc}(c_t)\big) \in \{0,1\}^{8t}.$$

3.1 Notations, Sets and Permutations

Let $\mathfrak{m}, \mathfrak{n}$ be arbitrary positive integers. We define the following sets and permutations:

- $\diamond$ $\mathsf{B}^2_{\mathfrak{m}}$: the set of all vectors in $\{0,1\}^{2\mathfrak{m}}$ whose Hamming weight is exactly $\mathfrak{m}$. Note that for $\mathbf{x} \in \mathbb{Z}^{2\mathfrak{m}}$ and $\psi \in \mathcal{S}_{2\mathfrak{m}}$ the following holds:

$$\begin{cases} \mathbf{x} \in \mathsf{B}^2_{\mathfrak{m}} \quad \Leftrightarrow \quad \psi(\mathbf{x}) \in \mathsf{B}^2_{\mathfrak{m}}; \\ \mathbf{x} \in \mathsf{B}^2_{\mathfrak{m}} \text{ and } \psi \xleftarrow{\$} \mathcal{S}_{2\mathfrak{m}}, \text{ then } \psi(\mathbf{x}) \text{ is uniform over } \mathsf{B}^2_{\mathfrak{m}}. \end{cases} \tag{2}$$

- $\diamond$ $T_{\psi,\mathfrak{n}}$, for $\psi \in \mathcal{S}_{\mathfrak{m}}$: the permutation that, when applied to a vector $\mathbf{v} = (\mathbf{v}_1\|\mathbf{v}_2\|\ldots\|\mathbf{v}_{\mathfrak{m}}) \in \mathbb{Z}^{\mathfrak{n}\mathfrak{m}}$, consisting of $\mathfrak{m}$ blocks of size $\mathfrak{n}$, re-arranges the blocks of $\mathbf{v}$ according to ψ, as follows,

$$T_{\psi,\mathfrak{n}}(\mathbf{v}) = (\mathbf{v}_{\psi(1)}\|\mathbf{v}_{\psi(2)}\|\ldots\|\mathbf{v}_{\psi(\mathfrak{n})}).$$

For convenience, when working with password alphabet $\Sigma_{\mathsf{all}} = \Sigma_D \cup \Sigma_S \cup \Sigma_L \cup \Sigma_U$ and password policy $f = \big((k_D, k_U, k_L, k_S), n_{\texttt{min}}, n_{\texttt{max}}\big)$, we introduce the following notations and sets:

- $\diamond$ $\eta_D = |\Sigma_D| = 10$, $\eta_S = |\Sigma_S| = 32$, $\eta_L = |\Sigma_L| = 26$, $\eta_U = |\Sigma_U| = 26$ and $\eta_{\mathsf{all}} = |\Sigma_{\mathsf{all}}| = 94$.
- $\diamond$ For $\alpha \in \{D, S, L, U, \mathsf{all}\}$: $\mathsf{Enc}_\alpha = \{\mathsf{enc}(w)) \mid w \in \Sigma_\alpha\}$.
- $\diamond$ SET_α for $\alpha \in \{D, S, L, U, \mathsf{all}\}$: the set of all vectors $\mathbf{v} = (\mathbf{v}_1\|\ldots\|\mathbf{v}_{\eta_\alpha}) \in \{0,1\}^{8\eta_\alpha}$, such that the blocks $\mathbf{v}_1, \ldots, \mathbf{v}_{\eta_\alpha} \in \{0,1\}^8$ are exactly the binary encodings of all characters in Σ_α, i.e.,

$$\big\{\mathbf{v}_1, \ldots, \mathbf{v}_{\eta_\alpha}\big\} = \big\{\mathsf{enc}(w)) : \ w \in \Sigma_\alpha\big\}.$$

- $\diamond$ $\mathsf{SET}_{n_{\mathsf{max}}}$: the set of all vectors $\mathbf{v} = (\mathbf{v}_1\|\ldots\|\mathbf{v}_{n_{\mathsf{max}}}) \in \{0,1\}^{n_{\mathsf{max}}\lceil\log n_{\mathsf{max}}\rceil}$, such that the blocks $\mathbf{v}_1, \ldots, \mathbf{v}_{n_{\mathsf{max}}} \in \{0,1\}^{\lceil\log n_{\mathsf{max}}\rceil}$ are exactly the binary decompositions of all integers in $[n_{\mathsf{max}}]$, i.e.,

$$\big\{\mathbf{v}_1, \ldots, \mathbf{v}_{n_{\mathsf{max}}}\big\} = \big\{\mathsf{bin}(1), \ldots, \mathsf{bin}(n_{\mathsf{max}})\big\}.$$

Observe that the following properties hold.

◇ For all $\alpha \in \{D, S, L, U, \mathsf{all}\}$, all $\mathbf{x} \in \mathbb{Z}^{8\eta_\alpha}$ and all $\psi \in \mathcal{S}_{\eta_\alpha}$:

$$\begin{cases} \mathbf{x} \in \mathsf{SET}_\alpha \quad \Leftrightarrow \quad T_{\psi,8}(\mathbf{x}) \in \mathsf{SET}_\alpha; \\ \mathbf{x} \in \mathsf{SET}_\alpha \text{ and } \psi \xleftarrow{\$} \mathcal{S}_{\eta_\alpha}, \text{ then } T_{\psi,8}(\mathbf{x}) \text{ is uniform over } \mathsf{SET}_\alpha. \end{cases} \tag{3}$$

◇ For all $\mathbf{x} \in \mathbb{Z}^{n_{\mathsf{max}}\lceil \log n_{\mathsf{max}} \rceil}$ and all $\psi \in \mathcal{S}_{n_{\mathsf{max}}}$:

$$\begin{cases} \mathbf{x} \in \mathsf{SET}_{n_{\mathsf{max}}} \quad \Leftrightarrow \quad T_{\psi,\lceil \log n_{\mathsf{max}} \rceil}(\mathbf{x}) \in \mathbf{x} \in \mathsf{SET}_{n_{\mathsf{max}}}; \\ \mathbf{x} \in \mathsf{SET}_{n_{\mathsf{max}}} \text{ and } \psi \xleftarrow{\$} \mathcal{S}_{n_{\mathsf{max}}}, \text{ then } T_{\psi,\lceil \log n_{\mathsf{max}} \rceil}(\mathbf{x}) \text{ is uniform over } \mathsf{SET}_{n_{\mathsf{max}}}. \end{cases} \tag{4}$$

3.2 Randomised Password Hashing from Lattices

We describe our randomised password hashing scheme $\mathcal{L}$ for passwords of length between two given integers n_{min} and n_{max}. At a high level, our scheme maps characters of the password pw to binary block vectors, re-arranges them with a random permutation χ, and finally computes the password hash as a KTX commitment ([28], see also Section 2.1) to a vector storing all the information on pw and χ. The scheme works as follows,

$\mathcal{L}.\mathsf{Setup}(\lambda)$**.** On input security parameter λ, the algorithm performs the following steps:

1. Choose parameters $n = \mathcal{O}(\lambda)$, prime modulus $q = \widetilde{\mathcal{O}}(n)$, and dimension $m = 2n\lceil \log q \rceil$.
2. Sample matrices $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times (n_{\mathsf{max}}\lceil \log n_{\mathsf{max}} \rceil + 8n_{\mathsf{max}})}$ and $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$.
3. Let the pre-hash salt space be $\mathbb{S}_P = \mathcal{S}_{n_{\mathsf{max}}}$ - the set of all permutations of n_{max} elements, and hash salt space be $\mathbb{S}_H = \{0,1\}^m$.
4. Output the public parameters $pp = \big(n, q, m, \mathbb{S}_P, \mathbb{S}_H, \mathbf{A}, \mathbf{B}\big)$.

$\mathcal{L}.\mathsf{PreSalt}(pp)$**.** Sample $\chi \xleftarrow{\$} \mathcal{S}_{n_{\mathsf{max}}}$ and output $s_P = \chi$.

$\mathcal{L}.\mathsf{PreHash}(pp, pw, s_P)$**.** Let $s_P = \chi \in \mathcal{S}_{n_{\mathsf{max}}}$ and $t \in [n_{\mathsf{min}}, n_{\mathsf{max}}]$ be the length of password pw. The pre-hash value P is computed as follows.

1. Compute $\mathsf{encode}(pw) \in \{0,1\}^{8t}$, consisting of t blocks of length 8.
2. Insert $n_{\mathsf{max}} - t$ blocks of length 8, each one being $\mathsf{enc}(g)$ for some non-printable ASCII character $g \in \Sigma_{\mathsf{ASCII}} \setminus \Sigma_{\mathsf{all}}$, into the block-vector $\mathsf{encode}(pw)$ to get $\mathbf{e} \in \{0,1\}^{8n_{\mathsf{max}}}$.[1]
3. Apply $T_{\chi,8}$ to get $\mathbf{e}' = T_{\chi,8}(\mathbf{e}) \in \{0,1\}^{8n_{\mathsf{max}}}$.
4. Output the pre-hash value $P = \mathbf{e}'$.

$\mathcal{L}.\mathsf{Salt}(pp)$**.** Sample $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$ and output $s_H = \mathbf{r}$.

$\mathcal{L}.\mathsf{Hash}(pp, P, s_P, s_H)$**.** Let $P = \mathbf{e}' \in \{0,1\}^{8n_{\mathsf{max}}}$, $s_P = \chi \in \mathcal{S}_{n_{\mathsf{max}}}$ and $s_H = \mathbf{r} \in \{0,1\}^m$. The hash value $\mathbf{h}$ is computed as follows,

1. Express the permutation χ as $\chi = [\chi(1), \ldots, \chi(n_{\mathsf{max}})]$, where for each $i \in [n_{\mathsf{max}}]$, $\chi(i) \in [n_{\mathsf{max}}]$. Then, form
$$\mathbf{e}_0 = \big(\mathsf{bin}(\chi(1) - 1)\| \ldots \|\mathsf{bin}(\chi(n_{\mathsf{max}}) - 1)\big) \in \{0,1\}^{n_{\mathsf{max}}\lceil \log n_{\mathsf{max}} \rceil}.$$
2. Form $\mathbf{x} = (\mathbf{e}_0 \| \mathbf{e}') \in \{0,1\}^{n_{\mathsf{max}}\lceil \log n_{\mathsf{max}} \rceil + 8n_{\mathsf{max}}}$ and output $\mathbf{h} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r} \in \mathbb{Z}_q^n$.

[1] This hides the actual length t of the password in the ZKPPC protocol in Section 3.4.

In the following theorem, we demonstrate that the proposed scheme satisfies the security requirements defined in Section 2.4.

Theorem 2. *Under the* SIS *assumption, the randomised password hashing scheme, $\mathcal{L}$, described above satisfies* 5 *requirements:* pre-image resistance*,* second pre-image resistance*,* pre-hash entropy preservation*,* entropy preservation *and* password hiding*.*

Proof. First, we remark that, by construction, if the pre-hash salt $s_P = \chi$ is given, then we can reverse the procedure used to extend the length t password by simply discarding any non-printable characters after applying the inverse of the permutation specified by s_P. Hence, if s_P is hidden, then due to its randomness, the min-entropy of P is larger than the min-entropy of pw. Thus, the proposed hashing scheme has the pre-hash entropy preservation and entropy preservation properties.

Next, note that $\mathbf{h} = \mathbf{A}\cdot\mathbf{x} + \mathbf{B}\cdot\mathbf{r} \bmod q$ is a proper KTX commitment of message $\mathbf{x}$ with randomness $\mathbf{r}$. Thus, from the statistical hiding property of the commitment scheme, the password hiding property holds.

Furthermore, if one can produce distinct pre-hash values P, P' that yield the same hash value $\mathbf{h}$, then one can use these values to break the computational binding property of the KTX commitment scheme. This implies that second pre-image resistance property holds under the SIS assumption.

Finally, over the randomness of matrix $\mathbf{A}$, password pw and pre-hash salt s_P, except for a negligible probability (i.e., in the event one accidentally finds a solution to the SIS problem associated with matrix $\mathbf{A}$), vector $\mathbf{A}\cdot\mathbf{x}$ accepts at least 2^β values in $\mathbb{Z}_q^n$, where β is the min-entropy of the dictionary `Dict` from which pw is chosen. Therefore, even if $\mathbf{A}\cdot\mathbf{x} = \mathbf{h} - \mathbf{B}\cdot s_H \bmod q$ is given, to find $P = \mathbf{e}'$, one has to perform 2^β invocations of $\mathsf{PreHash}$ which implies that the scheme satisfies the pre-image resistance property. $\square$

3.3 Techniques for Proving Set Membership

In our construction of ZKPPC in Section 3.4, we will have to prove that a linear relation of the form

$$\sum_i \big(\text{public matrix } \mathbf{M}_i\big)\cdot\big(\text{binary secret vector } \mathbf{s}_i\big) = \mathbf{h} \bmod q$$

holds, where each secret vector $\mathbf{s}_i$ must be an element of a given set of relatively small cardinality, e.g., $\mathsf{Enc}_D, \mathsf{Enc}_S, \mathsf{Enc}_L, \mathsf{Enc}_U, \mathsf{Enc}_{\mathsf{all}}$. Thus, we need to design suitable sub-protocols to prove set membership.

In the lattice-based world, a set membership argument system with logarithmic complexity in the cardinality of the set was proposed in [35], exploiting Stern-like protocols and Merkle hash trees. Despite its asymptotic efficiency, the actual efficiency is worse when the underlying set has small, constant size. To tackle the problems encountered here, we employ a different approach, which has linear complexity but is technically simpler and practically more efficient.

Suppose we have to prove that an $\mathfrak{n}$-dimensional vector $\mathbf{s}_i$ belongs to a set of $\mathfrak{m}$ vectors $\{\mathbf{v_1}, \ldots, \mathbf{v}_\mathfrak{m}\}$. To this end, we append $\mathfrak{m}-1$ blocks to vector $\mathbf{s}_i$ to get an $\mathfrak{nm}$-dimensional vector $\mathbf{s}_i^\star$ whose $\mathfrak{m}$ blocks are exactly elements of the set $\{\mathbf{v_1}, \ldots, \mathbf{v}_\mathfrak{m}\}$. At the same time, we append $\mathfrak{n}(\mathfrak{m}-1)$ zero-columns to public matrix $\mathbf{M}_i$ to get matrix $\mathbf{M}_i^\star$ satisfying $\mathbf{M}_i^\star\cdot\mathbf{s}_i^\star = \mathbf{M}_i\cdot\mathbf{s}_i$, so that we preserve the linear equation under consideration. In this way, we reduce the set-membership problem to the problem of proving the well-formedness of $\mathbf{s}_i^\star$. The latter can be done via random permutations of blocks in the framework of Stern's protocol. For instance, to prove that $\mathbf{s}_i \in \mathsf{Enc}_D$, i.e., $\mathbf{s}_i$ is a correct binary encoding of a digit, we extend it to $\mathbf{s}_i^\star \in \mathsf{SET}_D$, apply a random permutation to the extended vector, and make use of the properties observed in (3).

3.4 Zero-Knowledge Password Policy Check Protocol

We now present our construction of ZKPPC from lattices. Throughout, we use notations, sets and permutation techniques specified in Section 3.1 to reduce the statement to be proved to an instance of the relation $\mathrm{R}_{\mathrm{abstract}}$ considered in Section 2.2, which in turn can be handled by the Stern-like protocol of Figure 1.

Our protocol allows a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ in ZK that $\mathcal{P}$ knows a password pw that hashes to a given value with randomness $\chi, \mathbf{r}$, and satisfies some policy $f = \big((k_D, k_U, k_L, k_S), n_{\mathtt{min}}, n_{\mathtt{max}}\big)$.[2] Recall that $\mathcal{V}$ demands pw must have length between n_{min} and n_{max} inclusive, contain at least k_D digits, k_S symbols, k_L lower-case and k_U upper-case letters. For simplicity, we let $k_{\mathsf{all}} = n_{\mathsf{min}} - (k_D + k_U + k_L + k_S)$.

The common input consists of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times (n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil + 8 n_{\mathsf{max}})}$, $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, hash value $\mathbf{h} \in \mathbb{Z}_q^n$ and extra information

$$\varDelta = (\delta_{D,1}, \ldots, \delta_{D,k_D}, \delta_{S,1}, \ldots, \delta_{S,k_S}, \delta_{L,1}, \ldots, \delta_{L,k_L}, \delta_{U,1}, \ldots, \delta_{U,k_U}, \delta_{\mathsf{all},1}, \ldots, \delta_{\mathsf{all},k_{\mathsf{all}}}) \in [n_{\mathsf{max}}]^{n_{\mathsf{min}}},$$

which indicates the positions of the blocks, inside vector $P = \mathbf{e}'$, encoding k_D digits, k_S symbols, k_L lower-case letters, k_U upper-case letters and k_{all} other printable characters within pw. Revealing $\varDelta$ to $\mathcal{V}$ does not harm $\mathcal{P}$, since the original positions of those blocks (in vector $\mathbf{e}$) are protected by the secret permutation χ.

The prover's witness consists of vectors $\mathbf{x} = (\mathbf{e}_0 \| \mathbf{e}') \in \{0,1\}^{n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil + 8 n_{\mathsf{max}}}$ and $\mathbf{r} \in \{0,1\}^m$ satisfying the following conditions:

1. $\mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r} = \mathbf{h} \bmod q$;
2. $\mathbf{e}_0 = \big(\, \mathsf{bin}(\chi(1) - 1) \,\|\, \ldots \,\|\, \mathsf{bin}(\chi(n_{\mathsf{max}} - 1)) \,\big)$;
3. $\mathbf{e}'$ has the form $(\mathbf{x}_1, \ldots, \mathbf{x}_{n_{\mathsf{max}}})$, where, for all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and all $i \in [k_\alpha]$, it holds that $\mathbf{x}_{\delta_{\alpha,i}} \in \mathsf{Enc}_\alpha$.

We first observe that, if we express matrix $\mathbf{A}$ as $\mathbf{A} = \big[\mathbf{A}_0 \mid \mathbf{A}_1 \mid \ldots \mid \mathbf{A}_{n_{\mathsf{max}}}\big]$, where $\mathbf{A}_0 \in \mathbb{Z}_q^{n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil}$ and $\mathbf{A}_1, \ldots, \mathbf{A}_{n_{\mathsf{max}}} \in \mathbb{Z}_q^{n \times 8}$, then equation $\mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r} = \mathbf{h} \bmod q$ can be equivalently written as

$$\mathbf{A}_0 \cdot \mathbf{e}_0 + \sum_{\alpha \in \{D,S,L,U,\mathsf{all}\}, i \in [k_\alpha]} \mathbf{A}_{\delta_{\alpha,i}} \cdot \mathbf{x}_{\delta_{\alpha,i}} + \sum_{j \in [n_{\mathsf{max}}] \setminus \varDelta} \mathbf{A}_j \cdot \mathbf{x}_j + \mathbf{B} \cdot \mathbf{r} = \mathbf{h} \bmod q. \tag{5}$$

Note that, we have $\mathbf{e}_0 \in \mathsf{SET}_{n_{\mathsf{max}}}$. We next transform the witness vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{n_{\mathsf{max}}}, \mathbf{r}$ as follows,

- ◇ For all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and all $i \in [k_\alpha]$, to prove that $\mathbf{x}_{\delta_{\alpha,i}} \in \mathsf{Enc}_\alpha$, we append $\eta_\alpha - 1$ suitable blocks to $\mathbf{x}_{\delta_{\alpha,i}}$ to get vector $\mathbf{x}^\star_{\delta_{\alpha,i}} \in \mathsf{SET}_\alpha$.
- ◇ For vectors $\{\mathbf{x}_j\}_{j \in [n_{\mathsf{max}}] \setminus \varDelta}$, note that it is necessary and sufficient to prove that they are binary vectors (namely, they are encoding of characters that may or may not be printable). Similarly, we have to prove that $\mathbf{r}$ is a binary vector. To this end, we let $\mathbf{y} \in \{0,1\}^{8(n_{\mathsf{max}} - n_{\mathsf{min}})}$ be a concatenation of all $\{\mathbf{x}_j\}_{j \in [n_{\mathsf{max}}] \setminus \varDelta}$ and $\mathbf{z} = (\mathbf{y} \| \mathbf{r}) \in \{0,1\}^{8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m}$. Then, we append suitable binary entries to $\mathbf{z}$ to get $\mathbf{z}^\star \in \{0,1\}^{2(8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m)}$ with Hamming weight exactly $8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m$, i.e., $\mathbf{z}^\star \in \mathsf{B}^2_{8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m}$.

Having performed the above transformations, we construct the vector $\mathbf{w} \in \{0,1\}^\ell$, where

$$\ell = n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil + 8(k_D \eta_D + k_S \eta_S + k_L \eta_L + k_U \eta_U) + 8 k_{\mathsf{all}} \eta_{\mathsf{all}} + 2(8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m),$$

and $\mathbf{w}$ has the form:

[2] The construction we present considers the scenario where k_D, k_S, k_L, k_U are all positive. Our scheme can be easily adjusted to handle the case where one or more of them are 0.

$$\mathbf{w} = \big(\, \mathbf{e}_0 \,\|\, \mathbf{x}^\star_{\delta_{D,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{D,k_D}} \;\|\, \mathbf{x}^\star_{\delta_{S,1}} \,\|\, \ldots \|\mathbf{x}^\star_{\delta_{S,k_S}} \,\|\, \mathbf{x}^\star_{\delta_{L,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{L,k_L}} \,\|\, \mathbf{x}^\star_{\delta_{U,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{U,k_U}} \,\|\, \mathbf{x}^\star_{\delta_{\mathsf{all},1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{\mathsf{all},k_{\mathsf{all}}}} \,\|\, \mathbf{z}^\star \,\big). \tag{6}$$

When performing extensions over the secret vectors, we also append zero-columns to the public matrices in equation (5) so that it is preserved. Then, we concatenate the extended matrices to get $\mathbf{M} \in \mathbb{Z}_q^{n \times \ell}$ such that (5) becomes, with $\mathbf{v} = \mathbf{h} \in \mathbb{Z}_q^n$,

$$\mathbf{M} \cdot \mathbf{w} = \mathbf{v} \bmod q. \tag{7}$$

We have now established the first step towards reducing the given statement to an instance of the relation $\mathrm{R}_{\mathrm{abstract}}$ from Section 2.2. Next, we will specify the set VALID containing the vector $\mathbf{w}$, set $\mathcal{S}$ and permutations $\{\varGamma_\phi : \phi \in \mathcal{S}\}$ such that the conditions in (1) hold.

Define VALID as the set of all vectors $\mathbf{w} \in \{0,1\}^\ell$ having the form (6), where

- $\diamond$ $\mathbf{e}_0 \in \mathsf{SET}_{n_{\mathsf{max}}}$;
- $\diamond$ $\mathbf{x}^\star_{\delta_{\alpha,i}} \in \mathsf{SET}_\alpha$ for all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and all $i \in [k_\alpha]$;
- $\diamond$ $\mathbf{z}^\star \in \mathsf{B}^2_{8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m}$.

It can be seen that the vector $\mathbf{w}$ obtained above belongs to this tailored set VALID. Next, let us define the set of permutations $\mathcal{S}$ as follows,

$$\mathcal{S} = \mathcal{S}_{n_{\mathsf{max}}} \times \big(\mathcal{S}_{\eta_D}\big)^{k_D} \times \big(\mathcal{S}_{\eta_S}\big)^{k_S} \times \big(\mathcal{S}_{\eta_L}\big)^{k_L} \times \big(\mathcal{S}_{\eta_U}\big)^{k_U} \times \big(\mathcal{S}_{\eta_{\mathsf{all}}}\big)^{k_{\mathsf{all}}} \times \mathcal{S}_{2(8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m)}.$$

Then, for each element

$$\phi = \big(\pi,\, \tau_{D,1}, \ldots,\, \tau_{D,k_D},\, \tau_{S,1}, \ldots,\, \tau_{S,k_S},\, \tau_{L,1}, \ldots,\, \tau_{L,k_L},\, \tau_{U,1}, \ldots,\, \tau_{U,k_U},\, \tau_{\mathsf{all},1}, \ldots,\, \tau_{\mathsf{all},k_{\mathsf{all}}},\, \theta\big) \in \mathcal{S},$$

we define the permutation $\varGamma_\phi$ that, when applied to $\mathbf{w} \in \mathbb{Z}^\ell$ of the form

$$\mathbf{w} = \big(\, \mathbf{e}_0 \,\|\, \mathbf{x}^\star_{\delta_{D,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{D,k_D}} \;\|\, \mathbf{x}^\star_{\delta_{S,1}} \,\|\, \ldots \|\mathbf{x}^\star_{\delta_{S,k_S}} \,\|\, \mathbf{x}^\star_{\delta_{L,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{L,k_L}} \,\|\, \mathbf{x}^\star_{\delta_{U,1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{U,k_U}} \,\|\, \mathbf{x}^\star_{\delta_{\mathsf{all},1}} \,\|\, \ldots \,\|\, \mathbf{x}^\star_{\delta_{\mathsf{all},k_{\mathsf{all}}}} \,\|\, \mathbf{z}^\star \,\big).$$

where $\mathbf{e}_0 \in \mathbb{Z}^{n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil}$, $\mathbf{x}^\star_{\delta_{\alpha,i}} \in \mathbb{Z}^{8\eta_\alpha}$ for all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and $i \in k_\alpha$, and $\mathbf{z}^\star \in \mathbb{Z}^{2(8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m)}$, it transforms the blocks of vector $\mathbf{w}$ as follows,

- $\diamond$ $\mathbf{e}_0 \mapsto T_{\pi, \lceil \log n_{\mathsf{max}} \rceil}(\mathbf{e}_0)$.
- $\diamond$ For all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and all $i \in k_\alpha$: $\mathbf{x}^\star_{\delta_{\alpha,i}} \mapsto T_{\tau_{\alpha,i}, 8}(\mathbf{x}^\star_{\delta_{\alpha,i}})$.
- $\diamond$ $\mathbf{z}^\star \mapsto \theta(\mathbf{z}^\star)$.

Based on the properties observed in (2), (3), and (4), it can be seen that we have satisfied the conditions specified in (1), namely,

$$\begin{cases} \mathbf{w} \in \mathsf{VALID} \;\Leftrightarrow\; \varGamma_\phi(\mathbf{w}) \in \mathsf{VALID}, \\ \text{If } \mathbf{w} \in \mathsf{VALID} \text{ and } \phi \text{ is uniform in } \mathcal{S}, \text{ then } \varGamma_\phi(\mathbf{w}) \text{ is uniform in } \mathsf{VALID}. \end{cases}$$

Having reduced the considered statement to an instance of the relation $\mathrm{R}_{\mathrm{abstract}}$, let us now describe how our protocol is executed. The protocol uses the KTX string commitment scheme COM, which is statistically hiding and computationally binding under the SIS assumption. Prior to the interaction, the prover $\mathcal{P}$ and verifier $\mathcal{V}$ construct the matrix $\mathbf{M}$ and vector $\mathbf{v}$ based on the common inputs $(\mathbf{A}, \mathbf{B}, \mathbf{h}, \varDelta)$, while $\mathcal{P}$ builds the vector $\mathbf{w} \in \mathsf{VALID}$ from vectors $\mathbf{x}$ and $\mathbf{r}$, as discussed above. Then, $\mathcal{P}$ and $\mathcal{V}$ interact per Figure 1. We thus obtain the following result, as a corollary of Theorem 1.

Theorem 3. *Under the* SIS *assumption, the protocol above is a ZKPPC protocol with respect to the randomised password hashing scheme* $\mathcal{L}$ *from Section 3.2 and policy* $f = \big((k_D, k_U, k_L, k_S), n_{\mathtt{min}}, n_{\mathtt{max}}\big)$. *The protocol is a statistical ZKAoK with perfect completeness, soundness error* $2/3$ *and communication cost* $\mathcal{O}(\ell \log q)$.

Proof. Perfect completeness, soundness error $2/3$ and communication cost $\mathcal{O}(\ell \log q)$ of the protocol follow from the use of the abstract protocol in Figure 1. For simulation, we simply run the simulator of Theorem 1.

As for knowledge extraction, we first run the knowledge extractor of Theorem 1 to get the vector $\mathbf{w}' \in \mathsf{VALID}$ such that $\mathbf{M} \cdot \mathbf{w}' = \mathbf{v} \bmod q$. Then, we "backtrack" the transformations to extract from $\mathbf{w}'$, vectors $\mathbf{x}' = (\mathbf{e}'_0 \| \mathbf{x}'_1 \| \ldots \| \mathbf{x}'_{n_{\mathsf{max}}}) \in \{0,1\}^{n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil + 8 n_{\mathsf{max}}}$ and $\mathbf{r}' \in \{0,1\}^m$ such that

- ◇ $\mathbf{A} \cdot \mathbf{x}' + \mathbf{B} \cdot \mathbf{r}' = \mathbf{h} \bmod q$;
- ◇ $\mathbf{e}'_0 \in \mathsf{SET}_{n_{\mathsf{max}}}$;
- ◇ For all $\alpha \in \{D, S, L, U, \mathsf{all}\}$ and all $i \in k_\alpha$: $\mathbf{x}'_{\delta_{\alpha,i}} \in \mathsf{Enc}_\alpha$.

Notice that one can recover a permutation of n_{max} elements from an element of $\mathsf{SET}_{n_{\mathsf{max}}}$. Let χ' be the permutation encoded by $\mathbf{e}'_0$. Then, by applying the inverse permutation $T^{-1}_{\chi',8}$ to $(\mathbf{x}'_1 \| \ldots \| \mathbf{x}'_{n_{\mathsf{max}}})$, we recover $\mathbf{e}' \in \{0,1\}^{8 n_{\mathsf{max}}}$. Finally, by removing potential blocks of length 8 that correspond to encodings of non-printable ASCII characters from $\mathbf{e}'$, we obtain a vector that encodes some password string pw' satisfying policy f. □

Efficiency analysis. By inspection, we can see that, without using the big-O notation, each round of the proposed protocol has communication cost slightly larger than

$$\ell \log q = \big(n_{\mathsf{max}} \lceil \log n_{\mathsf{max}} \rceil + 8(k_D \eta_D + k_S \eta_S + k_L \eta_L + k_U \eta_U) + 8 k_{\mathsf{all}} \eta_{\mathsf{all}} + 2(8(n_{\mathsf{max}} - n_{\mathsf{min}}) + m)\big) \log q.$$

Let us estimate the cost in practice. Note that the KTX commitment scheme can work with relatively small lattice parameters, e.g., $n = 256$, $\log q = 10$, $m = 5120$. For a common password policy $f = \big((1,1,1,1), 8, 16\big)$, the communication cost would be about 17 KB. As each round has a soundness error of $2/3$, one may have to repeat the protocol many times in parallel to achieve a high level of confidence. For instance, if a soundness error of 2^{-30} is required, then one can repeat 52 times for a final cost of around 900 KB. In practical implementations, one can exploit various optimizations (e.g., instead of sending a random vector, one can send the PRNG seed used to generate it) to reduce the communication complexity.

4 Conclusion and Open Questions

Through the use of the KTX commitment scheme [28] and a Stern-like zero-knowledge argument of set membership, we designed a lattice-based zero-knowledge protocol for proving that a committed/hashed password sent to the server satisfies the required password policy. All together, we obtain the first ZKPPC that is based on the hardness of the SIS problem which to date remains quantum resistant. Unfortunately, there are no viable VPAKE protocols from lattices that can be coupled with our ZKPPC protocol to construct a complete privacy-preserving password-based authentication and key exchange system.

Our proposed ZKPPC protocol can be employed to securely register chosen passwords at remote servers with the following security guarantees: (1) Registered passwords are not disclosed to the server until used; (2) Each registered password provably conforms to the specified password policy. Although not being ready to be deployed in practice, we view this work as the first step in designing post-quantum privacy-preserving password-based authentication and key exchange systems.

We leave several open questions as potential future work: (1) to construct a more practical lattice-based ZKPPC; (2) to develop a lattice-based VPAKE; and (3) to extend lattice-based ZKPPC to other PAKE protocols, such as two-server PAKE, where the passwords are secretly shared between two servers, of which we assume at most one to be compromisable. The third question is similar to the one asked by Kiefer and Manulis [30] and as they noted, it is a challenge even in the classical discrete logarithm setting.

ACKNOWLEDGEMENTS. We would like to thank the anonymous reviewers of ISC 2017 for helpful comments. The research is supported by Singapore Ministry of Education under Research Grant MOE2016-T2-2-014(S) and by NTU under Tier 1 grant RG143/14.

References

1. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC 1996*. ACM, 1996.
2. C. Baum, I. Damgård, K. G. Larsen, and M. Nielsen. How to prove knowledge of small secrets. In *CRYPTO 2016*, volume 9816 of *Lecture Notes in Computer Science*. Springer, 2016.
3. C. Baum, I. Damgrd, S. Oechsner, and C. Peikert. Efficient commitments and zero-knowledge protocols from ring-sis with applications to lattice-based threshold cryptosystems. Cryptology ePrint Archive, Report 2016/997, 2016.
4. E. Bauman, Y. Lu, and Z. Lin. Half a century of practice: Who is still storing plaintext passwords? In *ISPEC 2015*, volume 9065 of *Lecture Notes in Computer Science*. Springer, 2015.
5. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*. Springer, 2000.
6. S. M. Bellovin and M. Merritt. Encrypted key exchange: password-based protocols secure against dictionary attacks. In *IEEE Symposium on Security and Privacy*. IEEE, 1992.
7. S. M. Bellovin and M. Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *ACM CCS 1993*. ACM, 1993.
8. F. Benhamouda, O. Blazy, C. Chevalier, D. Pointcheval, and D. Vergnaud. New techniques for sphfs and efficient one-round pake protocols. In *CRYPTO 2013*, volume 8402 of *Lecture Notes in Computer Science*. Springer, 2013.
9. F. Benhamouda, J. Camenisch, S. Krenn, V. Lyubashevsky, and G. Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *ASIACRYPT 2014*, volume 8873 of *Lecture Notes in Computer Science*. Springer, 2014.
10. F. Benhamouda, S. Krenn, V. Lyubashevsky, and K. Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In *Computer Security - ESORICS 2015, Part I*, volume 9326 of *Lecture Notes in Computer Science*. Springer, 2015.
11. F. Benhamouda and D. Pointcheval. Verifier-based password-authenticated key exchange: New models and constructions. Cryptology ePrint Archive, Report 2013/833, 2013.
12. R. Cramer, I. Damgård, C. Xing, and C. Yuan. Amortized complexity of zero-knowledge proofs revisited: Achieving linear soundness slack. In *EUROCRYPT 2017*, volume 10210 of *Lecture Notes in Computer Science*. Springer, 2017.
13. R. del Pino and V. Lyubashevsky. Amortization with fewer equations for proving knowledge of small secrets. Cryptology ePrint Archive, Report 2017/280, 2017.
14. Y. Ding and L. Fan. Efficient password-based authenticated key exchange from lattices. In *CIS 2011*. IEEE, 2011.
15. C. Dong, L. Chen, and Z. Wen. When private set intersection meets big data: an efficient and scalable protocol. In *ACM CCS 2013*. ACM, 2013.
16. C. Dong and F. Kiefer. Secure set-based policy checking and its application to password registration. In *CANS 2015*, volume 9476 of *Lecture Notes in Computer Science*. Springer, 2015.
17. D. Florêncio and C. Herley. Where do security policies come from? In *SOUPS 2010*. ACM, 2010.
18. J. Furukawa. Efficient and verifiable shuffling and shuffle-decryption. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 88-A(1):172–188, 2005.
19. S. Gates. Linkedin password hack: Check to see if yours was one of the 6.5 million leaked, 2012. `http://www.huffingtonpost.com/2012/06/07/linkedin-password-hack-check_n_1577184.html`.
20. R. Gennaro and Y. Lindell. A framework for password-based authenticated key exchange. In *EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.
21. C. Gentry, P. MacKenzie, and Z. Ramzan. A method for making password-based key exchange resilient to server compromise. In *CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*. Springer, 2006.
22. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC 2008*. ACM, 2008.

23. O. Goldreich and S. Goldwasser. On the limits of non-approximability of lattice problems. In *STOC 1998*. ACM, 1998.
24. J. Groth. Evaluating security of voting schemes in the universal composability framework. In *ACNS 2004*, volume 3089 of *Lecture Notes in Computer Science*. Springer, 2004.
25. T. Hunt. Have i been pwned, 2017. `https://haveibeenpwned.com/`, accessed on 7 July 2017.
26. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*. Springer, 2012.
27. J. Katz and V. Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In *ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*. Springer, 2009.
28. A. Kawachi, K. Tanaka, and K. Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*. Springer, 2008.
29. F. Kiefer. *Advancements in password-based cryptography.* PhD thesis, University of Surrey, 2016.
30. F. Kiefer and M. Manulis. Zero-knowledge password policy checks and verifier-based PAKE. In *ESORICS 2014*, volume 8713 of *Lecture Notes in Computer Science*. Springer, 2014.
31. F. Kiefer and M. Manulis. Blind password registration for two-server password authenticated key exchange and secret sharing protocols. In *ISC 2016*, volume 9866 of *Lecture Notes in Computer Science*. Springer, 2016.
32. F. Kiefer and M. Manulis. Blind password registration for verifier-based PAKE. In *AsiaPKC@AsiaCCS 2016*. ACM, 2016.
33. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *ASIACRYPT 2016*, volume 10032 of *Lecture Notes in Computer Science*. Springer, 2016.
34. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *ASIACRYPT 2016*, volume 10032 of *Lecture Notes in Computer Science*. Springer, 2016.
35. B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT 2016*, volume 9666 of *Lecture Notes in Computer Science*. Springer, 2016.
36. B. Libert, F. Mouhartem, and K. Nguyen. A lattice-based group signature scheme with message-dependent opening. In *ACNS 2016*, volume 9696 of *Lecture Notes in Computer Science*, 2016.
37. S. Ling, K. Nguyen, D. Stehlé, and H. Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*. Springer, 2013.
38. S. Ling, K. Nguyen, and H. Wang. Group signatures from lattices: Simpler, tighter, shorter, ring-based. In *PKC 2015*, volume 9020 of *Lecture Notes in Computer Science*. Springer, 2015.
39. S. Ling, K. Nguyen, H. Wang, and Y. Xu. Lattice-based group signatures: Achieving full dynamicity with ease. In *ACNS 2017*, volume 10355 of *Lecture Notes in Computer Science*, 2017.
40. V. Lyubashevsky. Lattice-based identification schemes secure under active attacks. In *PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*. Springer, 2008.
41. V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*. Springer, 2012.
42. D. Micciancio and C. Peikert. Hardness of SIS and LWE with small parameters. In *CRYPTO 2013*, volume 8402 of *Lecture Notes in Computer Science*. Springer, 2013.
43. D. Micciancio and S. P. Vadhan. Statistical zero-knowledge proofs with efficient provers: Lattice problems and more. In *CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*. Springer, 2003.
44. NIST. Post-quantum crypto standardization - call for proposals announcement, 2016. `http://csrc.nist.gov/groups/ST/post-quantum-crypto/cfp-announce-dec2016.html`.
45. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*. Springer, 1991.
46. C. Peikert and V. Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. In *CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*. Springer, 2008.
47. N. Perlroth. More than half a billion yahoo accounts have been hacked, yahoo confirms, 2016. `https://www.nytimes.com/2016/09/23/technology/yahoo-hackers.html`.
48. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005*. ACM, 2005.
49. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999.
50. J. Stern. A new paradigm for public key identification. *IEEE Trans. Information Theory*, 42(6):1757–1768, 1996.
51. B. Ur, P. G. Kelley, S. Komanduri, J. Lee, M. Maass, M. L. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, N. Christin, and L. F. Cranor. How does your password measure up? the effect of strength meters on password creation. In *USENIX Security Symposium 2012*. USENIX Association, 2012.

A Proof of Theorem 1

We provide the proof of Theorem 1 as it appears in [33] but first restate the theorem.

Theorem 1. *The protocol in Figure 1 is a statistical* ZKAoK *with perfect completeness, soundness error* $2/3$*, and communication cost* $\mathcal{O}(\ell \log q)$*. Namely:*

- *There exists a polynomial-time simulator that, on input* $(\mathbf{M}, \mathbf{v})$*, outputs an accepted transcript statistically close to that produced by the real prover.*
- *There exists a polynomial-time knowledge extractor that, on input a commitment* CMT *and* 3 *valid responses* $(\mathrm{RSP}_1, \mathrm{RSP}_2, \mathrm{RSP}_3)$ *to all* 3 *possible values of the challenge* Ch*, outputs* $\mathbf{w}' \in \mathsf{VALID}$ *such that* $\mathbf{M} \cdot \mathbf{w}' = \mathbf{v} \bmod q$*.*

Proof. Perfect completeness of the protocol can be checked: If a prover follows the protocol honestly, the verifier will always accept. It is also easy to see that the communication cost is bounded by $\mathcal{O}(\ell \log q)$.

We now prove that the protocol is a statistical zero-knowledge argument of knowledge.

Zero-Knowledge Property. We construct a PPT simulator SIM interacting with a (possibly dishonest) verifier $\widehat{\mathcal{V}}$, such that, given only the public inputs, with probability negligibly close to $2/3$, SIM outputs a simulated transcript that is statistically close to the one produced by the honest prover in the real interaction.

SIM first chooses uniformly at random, $\overline{Ch} \in \{1, 2, 3\}$, its prediction of Ch that $\widehat{\mathcal{V}}$ will *not* choose.

Case $\overline{Ch} = 1$: Using basic linear algebra over $\mathbb{Z}_q$, SIM computes a vector $\mathbf{w}' \in \mathbb{Z}_q^\ell$ such that $\mathbf{M} \cdot \mathbf{w}' = \mathbf{v} \bmod q$. Next, it samples $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^\ell$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Finally, it sends the commitment $\mathrm{CMT} = \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w; \rho_1), \quad C_2' = \mathsf{COM}(\varGamma_\phi(\mathbf{r}_w); \rho_2), \quad C_3' = \mathsf{COM}(\varGamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, the simulator responds as follows:

- If $Ch = 1$: Output $\perp$ and abort.
- If $Ch = 2$: Send $\mathrm{RSP} = \big(\phi, \mathbf{w}' + \mathbf{r}_w, \rho_1, \rho_3\big)$.
- If $Ch = 3$: Send $\mathrm{RSP} = \big(\phi, \mathbf{r}_w, \rho_1, \rho_2\big)$.

Case $\overline{Ch} = 2$: SIM samples $\mathbf{w}' \xleftarrow{\$} \mathsf{VALID}$, $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^\ell$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Then, it sends the commitment $\mathrm{CMT} = \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w; \rho_1), \quad C_2' = \mathsf{COM}(\varGamma_\phi(\mathbf{r}_w); \rho_2), \quad C_3' = \mathsf{COM}(\varGamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, the simulator responds as follows:

- If $Ch = 1$: Send $\mathrm{RSP} = \big(\varGamma_\phi(\mathbf{w}'), \varGamma_\phi(\mathbf{r}_w), \rho_2, \rho_3\big)$.
- If $Ch = 2$: Output $\perp$ and abort.
- If $Ch = 3$: Send $\mathrm{RSP} = \big(\phi, \mathbf{r}_w, \rho_1, \rho_2\big)$.

Case $\overline{Ch} = 3$: SIM samples $\mathbf{w}' \xleftarrow{\$} \mathsf{VALID}$, $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^\ell$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Then, it sends the commitment $\mathrm{CMT} = \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where $C_2' = \mathsf{COM}(\varGamma_\phi(\mathbf{r}_w); \rho_2)$, $C_3' = \mathsf{COM}(\varGamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3)$ as in the previous two cases, while

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot (\mathbf{w}' + \mathbf{r}_w) - \mathbf{v}; \rho_1).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, it responds as follows:

- If $Ch = 1$: Send RSP computed as in the case $(\overline{Ch} = 2, Ch = 1)$.
- If $Ch = 2$: Send RSP computed as in the case $(\overline{Ch} = 1, Ch = 2)$.
- If $Ch = 3$: Output $\perp$ and abort.

Observe that, in each of the cases considered above, since COM is statistically hiding, the distribution of the commitment CMT and challenge Ch from $\widehat{\mathcal{V}}$ are statistically close to those in the real interaction. Hence, the probability that the simulator outputs $\perp$ is negligibly close to $1/3$. Moreover, one can check that whenever the simulator does not halt, it will provide an accepted transcript, the distribution of which is statistically close to the prover's in the real interaction. In other words, the constructed simulator can successfully impersonate the honest prover with probability negligibly close to $2/3$.

Argument of Knowledge. Suppose $\mathrm{RSP}_1 = (\mathbf{t}_w, \mathbf{t}_r, \rho_2, \rho_3)$, $\mathrm{RSP}_2 = (\phi_2, \mathbf{w}_2, \rho_1, \rho_3)$ and $\mathrm{RSP}_3 = (\phi_3, \mathbf{w}_3, \rho_1, \rho_2)$ are 3 valid responses to the same commitment $\mathrm{CMT} = (C_1, C_2, C_3)$, with respect to all 3 possible values of the challenge. The validity of these responses implies that:

$$\begin{cases} \mathbf{t}_w \in \mathsf{VALID}; \\ C_1 = \mathsf{COM}(\phi_2, \mathbf{M} \cdot \mathbf{w}_2 - \mathbf{v} \bmod q; \rho_1) = \mathsf{COM}(\phi_3, \mathbf{M} \cdot \mathbf{w}_3; \rho_1); \\ C_2 = \mathsf{COM}(\mathbf{t}_r; \rho_2) = \mathsf{COM}(\mathit{\Gamma}_{\phi_3}(\mathbf{w}_3); \rho_2); \\ C_3 = \mathsf{COM}(\mathbf{t}_w + \mathbf{t}_r \bmod q; \rho_3) = \mathsf{COM}(\mathit{\Gamma}_{\phi_2}(\mathbf{w}_2); \rho_3). \end{cases}$$

Since COM is computationally binding, it implies that

$$\begin{cases} \mathbf{t}_w \in \mathsf{VALID};\ \phi_2 = \phi_3;\ \mathbf{t}_r = \mathit{\Gamma}_{\phi_3}(\mathbf{w}_3);\ \mathbf{t}_w + \mathbf{t}_r = \mathit{\Gamma}_{\phi_2}(\mathbf{w}_2) \bmod q; \\ \mathbf{M} \cdot \mathbf{w}_2 - \mathbf{v} = \mathbf{M} \cdot \mathbf{w}_3 \bmod q. \end{cases} \tag{8}$$

Since $\mathbf{t}_w \in \mathsf{VALID}$, if we let $\mathbf{w}' = [\mathit{\Gamma}_{\phi_2}]^{-1}(\mathbf{t}_w)$, then $\mathbf{w}' \in \mathsf{VALID}$. Furthermore, we have

$$\mathit{\Gamma}_{\phi_2}(\mathbf{w}') + \mathit{\Gamma}_{\phi_2}(\mathbf{w}_3) = \mathit{\Gamma}_{\phi_2}(\mathbf{w}_2) \bmod q,$$

which means that $\mathbf{w}' + \mathbf{w}_3 = \mathbf{w}_2 \bmod q$, and $\mathbf{M} \cdot \mathbf{w}' + \mathbf{M} \cdot \mathbf{w}_3 = \mathbf{M} \cdot \mathbf{w}_2 \bmod q$. As a result, we have $\mathbf{M} \cdot \mathbf{w}' = \mathbf{v} \bmod q$, concluding the proof. □