


LATKE: An identity-binding PAKE from lattice assumptions

Michael Rosenberg 
University of Maryland
micro@cs.umd.edu

March 5, 2023

Abstract

In a recent work, Cremers, Naor, Paz, and Ronen (CRYPTO '22) point out the problem of *catastrophic impersonation* in balanced password authenticated key exchange protocols (PAKEs). Namely, in a balanced PAKE, when a single party is compromised, the attacker learns the password and can subsequently impersonate *anyone to anyone* using the same password. The authors of the work present two solutions to this issue: CHIP, an *identity-binding PAKE* (iPAKE), and CRISP, a *strong identity-binding PAKE* (siPAKE). These constructions prevent the impersonation attack by generating a secret key on setup that is inextricably tied to the party's identity, and then *deleting the password*. Thus, upon compromise, all an attacker can immediately do is impersonate the victim. The strong variant goes further, preventing attackers from performing any precomputation before the compromise occurs.

In this work we present LATKE, an iPAKE from lattice assumptions in the random oracle model. In order to achieve security and correctness, we must make changes to CHIP's primitives, security models, and protocol structure.

Keywords: key agreement, password-based cryptography, IIoT, post-quantum cryptography

1 Introduction

A human-entered password is, for better and worse, one of the most common methods of authentication today. For some technologies, the human is there to enter the password every time, e.g., logging into a website, or connecting to a server via SSH. In these settings, neither party, neither the client nor the server, needs to store passwords for very long. However, there are settings and protocols wherein the participants have no choice but to store passwords for an indefinite term. Examples include WiFi equipment and nodes in a home or industrial mesh network. It is this setting we will concern ourselves with, as the consequences of device compromise are more severe.

Devices with shared long-term (possibly low-entropy) passwords need a way of establishing a high-entropy shared secret in order to facilitate secure communication. In a seminal work, Bellare and Merritt [BM92] describe a *password-authenticated key exchange* (PAKE)—a protocol in which two parties can use mutual knowledge of a password to establish a secure communication channel over an untrusted network. PAKE usage is widespread, being deployed Apple’s iCloud credential recovery system, in e-passports to secure the NFC protocol, and in the pairing protocol for WiFi networks [Hv22]. The protocols in these examples—namely SRP-6a [srp], PACE [CGIP12], and Dragonfly/SAE [Har15]—are *balanced* or *symmetric* PAKEs (often referred to simply as PAKEs). That is, any party in these protocols can initiate a session with any other party. Another type of PAKE, which we do not consider in this work, are *augmented*, or *asymmetric* PAKEs, which are designed for the scenario wherein one party is always a server and the other is always a client.

Binding to identity. A balanced PAKE is not the ideal primitive to use in a large network of peers who store passwords indefinitely. The authors of [CNPR22] point out that PAKEs have a *catastrophic impersonation* property—if a single device is compromised, the attacker recovers the password, and can use it to impersonate any device to any other device on the network. They present a solution to this issue with CHIP, an *identity-binding PAKE* (iPAKE), which replaces the on-device stored password with key material that is derived from the combination of the password with some identity information about the device (e.g., a simple identity string like “vacuum #5”). With this change, all an attacker is able to do from a compromised device is (1) impersonate the compromised device to other devices, or (2) mount a brute force attack to derive the password from the key material. The presentation of CHIP is fairly generic: it is phrased as the composition of an identity-based key agreement protocol (IBKA) with an ordinary PAKE. The iPAKE is proven secure in the Universal Composability (UC) model [Can01] using the authors’ (novel) definition of iPAKE functionality. The same paper also introduces CRISP, a *strong* iPAKE (siPAKE), which frustrates attack type (2) by making precomputation infeasible. With CRISP, an attacker can only begin their brute-force search once a compromise has already happened.

Need for post-quantum constructions. Being built from Diffie-Hellman-type assumptions, the CHIP and CRISP constructions are extremely efficient, both in computation and communication cost. But as the horizon for cryptographically relevant quantum computers becomes shorter [You22], it is important to have readily available alternatives which rely on post-quantum assumptions. This is especially relevant in systems which handle data that is intended to be confidential for a long period of time, and in systems which have a long migration or updating period for their devices and firmware, as is common in industrial settings [Pau22]. As of this writing, to the authors’ knowledge, no iPAKE from post-quantum assumptions has been constructed.

Our contribution. In this work, we present Lattice password-Authenticated identity-Tied Key Exchange (LATKE), an iPAKE from lattice assumptions in the random oracle model, built using the CHIP/CRISP framework.

We will begin the construction with an existing lattice-based IBKA and iteratively tweak it until it satisfies the properties required by the CHIP framework (Section 3). Interestingly, *these tweaks will break both security and correctness of the IBKA*. We call the resulting scheme an *output-fuzzy IBKA* (OF-IBKA)—a key agreement protocol in which the two copies of the shared secret in any honest execution are *approximately* but not exactly equal. We will patch the security issue by identifying a model that is weaker than the one used by CHIP’s IBKA [FG10], but still sufficient for iPAKE security. We will patch the correctness issue by replacing the ordinary PAKE functionality with a *fuzzy PAKE* [DHP⁺18], i.e., a PAKE that which can tolerate passwords that differ up to some distance in some metric (Section 4). We prove the OF-IBKA secure in the new model, and show that the final construction realizes the iPAKE functionality in the UC model.

Finally, we present several avenues for future work (Section 5), including a new fPAKE which we posit is secure in the BPR model [BPR00] under the Pairing with Errors (PWE) assumption (reducible from Module-LWE). This new fPAKE is significantly more efficient than that of [DHP⁺18], though it is likely not possible to prove UC-secure. We conclude with the question of what security guarantees can be made about an iPAKE outside of the UC model.

1.1 CHIP overview

CHIP [CNPR22] offers an appealing generic construction for an iPAKE, requiring just an ordinary PAKE and an identity-based key agreement protocol (IBKA). The construction uses the IBKA in a novel way, not as a trusted third party, but as a mechanism to generate a public key and issue *oneself* an identity-bound secret key. In the framework, this is called a *password file*—a value derived from the password, but bound to a specific party’s identity. In the case of CHIP, the password file is an IBKA user secret key `usk`, along with the main public key `mpk`.

On setup, a party with password `pw` and identifier `id` generates the main keypair by executing the `KeyGen` procedure of an IBKA *using* $H(\text{pw})$ *as the random coins*. The party then generates `usk` by executing the `Extract` procedure of the IBKA on `id`, using knowledge the main secret key. With the password file in hand, the party then deletes all intermediate values, *including the password*. In the online part of CHIP, two parties run the key agreement protocol using the knowledge of their respective password files, and use the result as the *input* to an ordinary PAKE protocol. The result of the PAKE is the final shared secret. This protocol is illustrated in Figure 1.

The primary purpose of composing with an ordinary PAKE is forward secrecy—if the IBKA is not itself forward-secret, composing with a PAKE will prevent an attacker who learns `pw` from going back through recorded (possibly interfered-with) CHIP transcripts and deriving the session key. This structure is helpful in our use case. It implies that it is possible to use weak (non-forward-secret, non-CCA-secure) IBKA schemes without compromising security of the larger scheme, since parties will always pass the output through a PAKE.

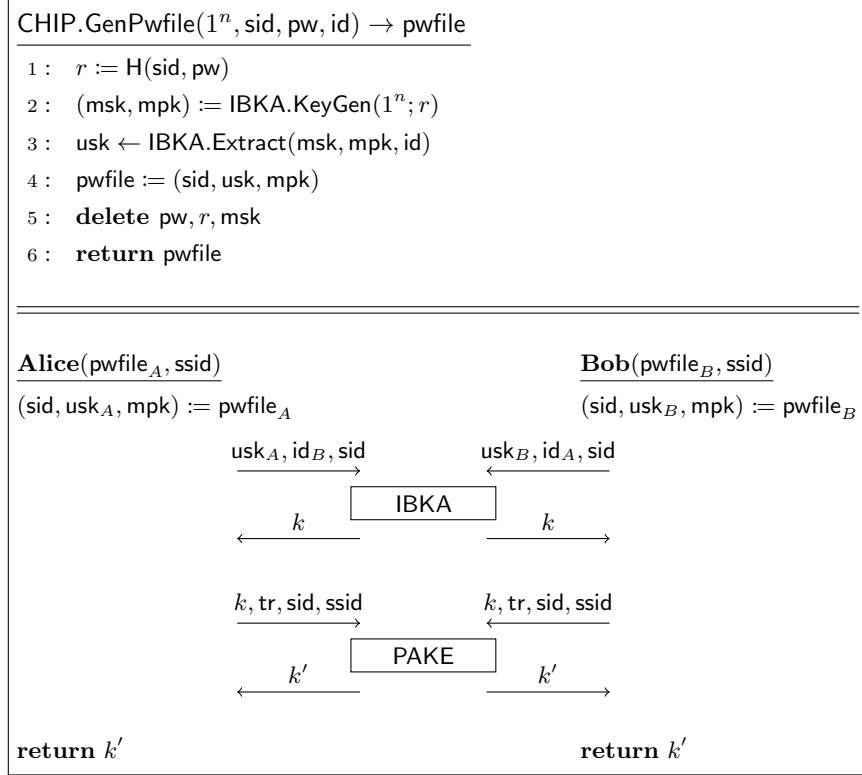


Figure 1: An outline of the CHIP protocol. IBKA and PAKE are generic identity-based key agreement and password-authenticated key exchange algorithms, respectively. tr represents the protocol transcript up to that point. sid and ssid are session identifiers, as defined in Section 2.8.

IBKA requirements. CHIP requires its IBKA to satisfy two properties. First, it must be resistant to *key compromise impersonation* (KCI) attacks (full security definition in Section 2.7). That is, if Mallory compromises Alice, then Mallory should not be able to do anything beyond impersonate Alice to other parties. Importantly, Mallory should not be able to impersonate Bob to Alice, or worse, impersonate Bob to Charlie. Second, to prevent brute-forcing, the IBKA must have a message flow that is *independent* of the random coins r used in KeyGen (here, $\text{H}(\text{pw})$). That is, an adversary should not be able to use the messages exchanged in the IBKA protocol to tell whether a particular r was used. We call this *keygen coin independence* (Section 2.4).

Our goal is to construct an identity-binding PAKE in the model of CHIP, using lattice-based cryptographic assumptions. Since (structured) lattice-based PAKEs already exist [KV09, GK10, CDVW12, DAL⁺17,

BBDQ18, JGH⁺20, RG22], it appears the only question is in how to construct an IBKA with the above properties. However, our answer to this will directly constrain our choice of PAKE, yielding a construction that differs slightly from the original CHIP blueprint.

2 Preliminaries

In this section we present the notation, definitions, and security models necessary for the construction of LATKE. Among these are some novel definitions, namely those of an *output-fuzzy IBKA* and *keygen coin independence*, which we require for later theorem statements.

2.1 Notation

We write \mathbb{Z}_q to represent $\mathbb{Z}/q\mathbb{Z}$ for a non-negative integer q . Matrices are written in bold uppercase, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and vectors are written in bold lowercase $\mathbf{v} \in \mathbb{Z}_q^n$. The transpose of \mathbf{A} and \mathbf{v} are denoted by \mathbf{A}^T and \mathbf{v}^T , respectively. The components of a vector are denoted with subscripts, $\mathbf{v} = (v_1, \dots, v_n)$. For $\mathbf{v} \in \mathbb{Z}_q^n$, we define $\|\mathbf{v}\|_p$ to be the ℓ_p norm of $\bar{\mathbf{v}} \in \mathbb{Z}^n$, where each \bar{v}_i is the representative of v in $\{-\lfloor q/2 \rfloor, \dots, \lceil q/2 \rceil\}$.

We write non-deterministic algorithms as $\text{Alg}(x; r)$ where x denotes its input, and r denotes its random coins. We write $x \leftarrow \mathcal{S}$ to denote sampling a value from the probability distribution, denoted by calligraphic letters, \mathcal{S} , and $x \leftarrow_{\$} S$ to denote sampling a value uniformly from the set S . For zero-knowledge proof systems, we write relations as $R = \{(x; w) : P(x, w)\}$, where x is the *instance*, w is the *witness*, and P is some efficiently computable predicate. For our security proofs we will consider probabilistic polynomial time (PPT) adversaries, and denote them with calligraphic letters \mathcal{A} . We denote the output x, x' of either side of an interactive protocol between parties A, B by $(x, x') \leftarrow (A \leftrightarrow B)$. We use n to denote the security parameter.

We use little- ω notation, $f(n) \in \omega(g(n))$ for $f, g : \mathbb{N} \rightarrow \mathbb{R}$ to mean that f dominates g asymptotically. That is, for any $c > 0$, there is an n_0 such that $f(n) > cg(n)$ for all $n > n_0$. We say a function $f : \mathbb{N} \rightarrow \mathbb{R}$ is *negligible in n* , denoted $f(n) = \text{negl}(n)$, iff $|f(n)| = n^{-\omega(1)}$ or, equivalently, for any $c > 0$, there is an n_0 such that $|f(n)| < n^{-c}$ for all $n > n_0$. We say a function $f : \mathbb{N} \rightarrow \mathbb{R}$ is *polynomial in n* , denoted $f(n) = \text{poly}(n)$, iff for some $M, d > 0$, there is an n_0 such that $|f(n)| \leq Mn^c$ for all $n > n_0$. We say a $f : \mathbb{N} \rightarrow \mathbb{R}$ is *noticeable in n* , iff for some $M, d > 0$, there is an n_0 such that $|f(n)| \geq M/n^c$ for all $n > n_0$.

2.2 Probability

A *probability ensemble* \mathcal{S} is an infinite sequence of probability distributions $\mathcal{S}_1, \mathcal{S}_2, \dots$, where each \mathcal{S}_i is over a set $A_i \subset \{0, 1\}^{\ell(i)}$, where $\ell(n) = \text{poly}(n)$ is some length function. The *statistical distance* between two probability distributions $\mathcal{S}, \mathcal{S}'$ over sets A, A' is

$$\Delta(\mathcal{S}, \mathcal{S}') = \frac{1}{2} \sum_{x \in A \cup A'} |\Pr[y = x \mid y \leftarrow \mathcal{S}] - \Pr[y = x \mid y \leftarrow \mathcal{S}']|.$$

We say that two probability ensembles $\mathcal{S}, \mathcal{S}'$ are *statistically indistinguishable*, denoted $\mathcal{S} \approx \mathcal{S}'$ iff $\Delta(\mathcal{S}_n, \mathcal{S}'_n) = \text{negl}(n)$. We say that two probability ensembles $\mathcal{S}, \mathcal{S}'$ are *computationally indistinguishable* under the P-hardness assumption, denoted $\mathcal{S} \stackrel{c}{\approx} \mathcal{S}'$, iff, for any PPT distinguisher \mathcal{D} with noticeable success probability—i.e., $|\Pr[\mathcal{D}(\mathcal{S}) = 0] - \Pr[\mathcal{D}(\mathcal{S}') = 0]|$ is noticeable—there exists a PPT adversary \mathcal{B} against the P problem with noticeable success probability. We say that an event X in some probability ensemble \mathcal{S} occurs with *overwhelming probability* iff $\Pr[X] = 1 - f(n)$ where $f(n) = \text{negl}(n)$.

2.3 PAKEs and iPAKEs

PAKE definition. We describe the function of a PAKE and its intended security properties. A (*balanced*) *password-authenticated key exchange protocol*, is a two-party key exchange protocol wherein both parties use mutual knowledge of a low-entropy password pw to establish a high-entropy *shared session key*. While there is no single security definition [BPR00, CHK⁺05, AHH21, RX22], the security goal for a PAKE is to (1) prevent passive adversaries (i.e., eavesdroppers) from learning anything about the password, and (2) limit active adversaries to one password guess per protocol instance *even if given access to session keys*.

An instructive example of an insecure PAKE: the parties perform an unauthenticated Diffie-Hellman key exchange to derive shared secret ss , then let the session key be $k = \text{KDF}(\text{pw}, \text{ss})$. While this scheme satisfies goal (1), it fails goal (2). A meddler-in-the-middle (MitM) adversary can insert themselves into the key exchange to derive ss . Then, if the adversary learns k at any point, it can brute force the password offline by simply re-running the KDF with different pw until its output equals k . A common technique that secure PAKEs use to prevent this attack is to *blind* the Diffie-Hellman key shares by multiplying by a password-derived group element [KOY01, BG03, JG04, AP05].

Authentication. PAKEs are either *explicitly* or *implicitly authenticated*. A PAKE is *explicitly authenticated* if both parties know at the end of the protocol whether the protocol succeeded or failed, i.e., whether they have established a shared session key with an honest party or not. A PAKE is *implicitly authenticated* if the parties do not know whether the protocol succeeded or failed. Frequently, it suffices to consider just implicitly authenticated PAKEs. One generic way of adding explicit authentication to an implicit PAKE whose output is a hash is to perform the final hash twice, with domain-separated hash functions, and use one of them to compute and share a MAC of a fixed string. That is, if the final key is $k = \text{H}(x)$, then $k' := \text{H}'(x)$ and the *confirmation tag* of party $i \in \{0, 1\}$ is computed as $\text{MAC}_{k'}(i)$ and shared with party $1 - i$ [Kra05, Section 8].

Catastrophic impersonation in PAKEs. Balanced PAKEs are not very robust to corruption. They require all parties to store a copy of the password indefinitely. Thus, an attacker who *corrupts* a party, i.e., obtains their long-term secrets, will learn pw and be able to impersonate *any* party with that password. Since all the PAKE authentication property does is ensure the two parties know the same password, there is no way to detect

the imposter.

iPAKE definition. The notion of *identity-binding PAKE* (iPAKE) was first introduced in [CNPR22] and instantiated as CHIP. For the full simulation security definition, we refer the reader to the original paper. In words, the purpose of the iPAKE construction is to serve as a PAKE which mitigates the catastrophic impersonation scenario described above. Rather than store the password indefinitely, parties in an iPAKE protocol run a one-time procedure `GenPwfile` on setup, wherein they supply an identifier string of their choosing (e.g., “Andrew’s smartwatch”) and the password. The procedure outputs a new secret, called a *password file* `pwfile`, which is inextricably bound to the identifier string. The party then stores `pwfile` and *deletes* `pw`.

In the future, if a party, Alice, is compromised and her device contents is dumped, the attacker only learns `pwfile`. With this, the attacker can impersonate Alice to anyone, but it cannot impersonate Bob to Alice, nor Bob to Charlie. The attacker can mount a brute force attack against `pwfile` to derive `pw`, but this is already significantly more computation than was required in the PAKE instance, especially if $H(\text{pw})$ is computed with a time- and memory-intensive password hashing function, such as `scrypt` [Per09], `Argon2` [BDK15], or `Balloon` [BCS16].

2.4 Identity-based key agreement protocol

We define the structure of a key agreement protocol as in [CK01, Kra05, FG10], adapted to an identity-based setting.

An identity-based key agreement protocol (IBKA) is an interactive protocol wherein pairs of *parties* communicate to derive a *shared secret* `ss`. Each party has a unique public ID. The `KeyGen` procedure (possibly executed by a trusted third party) establishes the public parameters for the protocol as well as the *main secret key* `msk` and *main public key* `mpk`. The `Extract` procedure, which is possibly non-deterministic, takes an identifier string `id` and uses knowledge of `msk` to issue a *user secret key* `usk`. A party with `id` and a corresponding `usk` can then participate in the key agreement protocol.

Sessions. An instance of the protocol is called a *session*, defined by the quadruple $(\text{id}_A, \text{id}_B, \text{tr}_{\text{out}}, \text{tr}_{\text{in}})$, where `trout` is the transcript of outgoing messages, and `trin` is the transcript of incoming messages. The two parties participating in a session, here `idA` and `idB`, are called *peers*. Two sessions with swapped IDs and transcripts as said to be *matching*, i.e., $(\text{id}_A, \text{id}_B, \text{tr}_{\text{out}}, \text{tr}_{\text{in}})$ and $(\text{id}_B, \text{id}_A, \text{tr}_{\text{in}}, \text{tr}_{\text{out}})$ are matching sessions (in other words, they are the same session, just from different perspectives).

Each participant maintains the *session state*, which includes the running incoming and outgoing transcripts, as well as all random coins. Once the protocol terminates, the session is *completed*, and the party will output its shared secret and delete the session state. Note that a completed session might have an incomplete matching session. Sessions may also be *aborted*, meaning the party stops the protocol and outputs no shared secret.

Properties. We will informally use the term *authenticity* to refer to the property that a party in a secure IBKA is ensured that its session’s shared

secret is only known to its peer. This is encompassed by the definition of SK-security in Section 2.7.

Due to the details of LATKE’s construction, we will not be able to show perfect or even statistical correctness, i.e., that both parties in an honest execution of the protocol will derive the same shared secret with overwhelming probability. Instead, we will use a notion of approximate correctness. We call the new construction an *output-fuzzy IBKA*.

Definition (Output-fuzzy identity-based key agreement (OF-IBKA)). *An identity-based key agreement protocol is called output-fuzzy with d -distance δ iff, for any honest execution of the protocol, the shared secret of the two parties are within distance δ of each other under the distance function d , with overwhelming probability. Formally, for any pair of parties $(\text{id}_A, \text{id}_B)$,*

$$\Pr \left[d(\text{ss}, \text{ss}') > \delta \mid \begin{array}{l} (\text{msk}, \text{mpk}) \leftarrow \text{KeyGen}(1^n) \\ \text{usk}_A \leftarrow \text{Extract}(\text{msk}, \text{id}_A) \\ \text{usk}_B \leftarrow \text{Extract}(\text{msk}, \text{id}_B) \\ (\text{ss}, \text{ss}') \leftarrow (A(\text{usk}_A) \Leftrightarrow B(\text{usk}_B)) \end{array} \right] = \text{negl}(n).$$

We will also need a way of hiding the random coins of our `KeyGen` procedure from any passive adversary, since the CHIP construction uses $H(\text{pw})$ as the random coins. This is captured by the following definition, first stated informally in [CNPR22].

Definition (Keygen coin independence). *An IBKA is keygen coin independent iff protocol transcripts reveal nothing about the random coins used in the `KeyGen` procedure. Formally, consider the following experiment, given a PPT adversary \mathcal{A} : sample distinct $r_0, r_1 \leftarrow_{\$} \{0, 1\}^n$, pick $b \leftarrow_{\$} \{0, 1\}$, let $(\text{msk}, \text{mpk}) \leftarrow \text{KeyGen}(1^n; r_0)$, and let $\{\text{tr}_i\}$ be a $\text{poly}(n)$ -sized set of IBKA execution transcripts between honest parties identified by $\{\text{id}_i\}$. The IBKA is keygen coin independent iff for any PPT adversary \mathcal{A} , and for all choices of $\{\text{id}_i\}$,*

$$\text{Adv}_{\mathcal{A}}^{\text{kgci}}(n) = \left| \frac{1}{2} - \Pr[\mathcal{A}(r_b, \{\text{tr}_i\}, \{\text{id}_i\}) = b] \right| = \text{negl}(n)$$

Remark 1. If the goal is, as in a PAKE, to limit the adversary’s number of password guesses (here, guesses at r_0), to one per protocol execution, then the above definition is too weak to use in general. The definition only considers passive adversaries, and it is fathomable that an active adversary might obtain a significant advantage in this game. The reason this definition is sufficient for our (and CHIP’s) use case is because both protocols have message flows which do not depend on incoming messages. That is, it is not possible for an active adversary to make an honest party behave differently (other than aborting due to a bad zero-knowledge proof, which we discuss below). Protocols without this property will have to use a definition that captures adversary capabilities similar to those in the nrSK-security game (Section 3.3.1).

Remark 2. We contrast keygen coin independence with *key escrow freeness*, as used in [BCGP08]. This property requires that a passive adversary with knowledge of `msk` cannot derive shared secret. This is distinct from keygen coin independence. In the latter property, a passive

adversary is allowed to derive shared secrets, but cannot know when it succeeded, i.e., it cannot know whether its `msk` guess is correct. This may seem like a strictly weaker requirement than key escrow freeness, but it is entirely separate; there are key-escrow-free protocols [BCGP08, Protocol 2] which are not keygen coin independent due to their generic use of a KEM. The lack of keygen coin independence in KEM-based schemes is discussed in Section 3.1.

2.5 Zero-knowledge proofs

We will require a noninteractive zero-knowledge proof-of-knowledge (NIZK) scheme for our IBKA construction. Informally, a NIZK is a two-round protocol wherein a *prover* convinces a *verifier* of some efficiently checkable statement. Often the statement will have some information that the prover wants to keep hidden. This, as a collection, is called the statement’s *witness*. The public parts of the statement are called the *instance*. For example, in the statement “I know a set of assignments for the variables x_1, \dots, x_n such that the boolean circuit $C(x_1, \dots, x_n) = 1$ ”, the set of assignments is the witness, and the circuit C (and its depth, and the input size n) is the instance.

NIZK functionality. For formal definitions of a zero-knowledge proof-of-knowledge protocols, we refer the reader to [Tha23]. For our presentation, it will suffice to explain the procedure the scheme exposes, as well as the properties they satisfy.

`Setup($1^n, R$)` \rightarrow (`pk`, `vk`) Constructs the *proving* and *verifying* keys for the relation R .

`Prove(pk, x, w)` \rightarrow π Constructs a zero-knowledge proof of the given relation with respect to instance x and witness w .

`Vfy(vk, x, π)` \rightarrow $\{0, 1\}$ Verifies the proof π of the given relation with respect to the instance x . Returns 1 on success.

Correctness. The property of *correctness* implies that valid proofs pass verification with overwhelming probability. That is, if $(x, w) \in R$,

$$\Pr \left[\text{Vfy}(\text{vk}, x, \pi) = 1 \mid \begin{array}{l} (\text{pk}, \text{vk}) \leftarrow \text{Setup}(1^n, R) \\ \pi \leftarrow \text{Prove}(\text{pk}, x, w) \end{array} \right]$$

is overwhelming.

Soundness. The property of *soundness* implies the existence of an *extractor* for a proof system. An extractor $\text{Ext}(x, \pi)$ for the relation R is an efficient algorithm which, given a valid proof, will return the witness used to generate that proof. Obviously the NIZK scheme would be insecure if the extractor worked in any context. In reality, the extractor needs extra help, e.g., by having access to the prover and repeatedly rewinding it, by operating in the algebraic group model and extracting from algebraic provers, or by operating in the random oracle model and programming the random oracle as needed.

Zero-knowledge. The property of *zero-knowledge* implies the existence of a *simulator* for a proof system. A simulator $\text{Sim}(x)$ is an efficient

algorithm which, given an instance, will produce a valid proof π which is distributed identically to the honest proofs generated by $\text{Prove}(\text{pk}, x, w)$, for any valid w . Again, a NIZK scheme would be insecure if a simulator worked in any context. To function, the simulator needs extra help, e.g., by knowing some trapdoor information about the proving key or by being able to program the random oracle.

For the sake of concreteness in the LATKE construction, we may consider the NIZK presented in [LNP22], which is proven sound and zero-knowledge in the random oracle model from (Module-)LWE and (Module-)SIS hardness assumptions. The relation we use in the IBKA, R_{SIS} (Section 3.2), is extremely efficient to prove in this system.

2.6 Lattice definitions and lemmas

Here we define the structures, hardness assumptions, and lemmas we will need for LATKE.

Discrete Gaussian distribution. An n -dimensional lattice Λ is a discrete subgroup of \mathbb{R}^n . The *Gaussian function* ρ_s on \mathbb{R}^n with parameter $s \in \mathbb{R}^+$, centered at $\mathbf{c} \in \mathbb{R}^n$ is defined as

$$\rho_{s,\mathbf{c}}(\mathbf{x}) := \exp(-\pi \|\mathbf{x} - \mathbf{c}\|_2^2 / s^2)$$

The *discrete Gaussian distribution* $\mathcal{D}_{\Lambda,s,\mathbf{c}}$ on an n -dimensional lattice Λ with parameter $s \in \mathbb{R}^+$, centered at $\mathbf{c} \in \mathbb{R}^n$ is defined to be the distribution with probability density function

$$\mathcal{D}_{\Lambda,s,\mathbf{c}}(\mathbf{x}) := \frac{\rho_{s,\mathbf{c}}(\mathbf{x})}{\sum_{\mathbf{y} \in \Lambda} \rho_{s,\mathbf{c}}(\mathbf{y})}.$$

For brevity, we will omit the subscript \mathbf{c} if $\mathbf{c} = 0$.

Learning with errors. For a probability distribution χ on \mathbb{Z}_q , the *Learning with Errors (LWE) distribution* $A_{q,n,m,\chi}$ is sampled by selecting $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and $\mathbf{e} \leftarrow \chi^m$, and outputting $(\mathbf{A}, \mathbf{A}^T \mathbf{s} + \mathbf{e})$.

The *LWE problem*, denoted $\text{LWE}_{q,n,m,\chi}$ is to distinguish a sample from $A_{q,n,m,\chi}$ from a sample from the uniform distribution on $(\mathbb{Z}_q^{n \times m}, \mathbb{Z}_q^m)$.

Preimage sampleable functions. We use the definitions from [GPV08]. A *preimage sampleable function* (PSF) scheme $(\text{GenTrap}, \text{SampleDom}, \text{SamplePre})$ is a tuple of procedures over domain D_n and range R_n with an efficiently computable function family $f_a : D_n \rightarrow R_n$ such that:

$\text{GenTrap}(1^n) \rightarrow (a, t)$ Generates an *instance* a , which is the description of some function in the function family f_a . Also generates secret *trapdoor* t , which is used for preimage sampling.

$\text{SampleDom}(1^n) \rightarrow x$ Samples an element from a (possibly non-uniform) distribution on D_n , for which the distribution of $f_a(x)$ is (statistically close to) uniform over R_n .

$\text{SamplePre}(a, t, y) \rightarrow x$ Given a target $y \in R_n$, uses the trapdoor information to sample an element from (a distribution statistically close to) the conditional distribution $x \leftarrow \text{SampleDom}(1^n)$, given $f_a(x) = y$.

In this work, we will only consider *linear* PSFs, i.e., PSFs of the form described in [GPV08, MP12], where $f_{\mathbf{A}}(\mathbf{E}) = \mathbf{A}\mathbf{E}$ for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{E} \in \mathbb{Z}_q^{m \times \ell}$, for integers q, m, n, ℓ , where SampleDom is $\mathcal{D}_{\mathbb{Z}^m, s}$ for some $s > 0$, and where the instance \mathbf{A} returned by GenTrap is statistically close to the uniform distribution. In the multi-bit encryption setting, when $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_\ell]$, we use the notation

$$\text{SamplePre}(\mathbf{U}) := [\text{SamplePre}(\mathbf{u}_1), \dots, \text{SamplePre}(\mathbf{u}_\ell)],$$

so that if $\mathbf{E} \leftarrow \text{SamplePre}(\mathbf{U})$, then $\mathbf{A}\mathbf{E} = \mathbf{U}$.

In some theorems, we will also impose a min-entropy requirement on the output distribution of SamplePre . A useful bound is given in the following lemma. In practice, the “sufficiently large” clause is satisfied by n far smaller than the security parameter.

Lemma 1 (Adapted from [GPV08, Lemma 2.10]). *For any n -dimensional lattice Λ with sufficiently large n , $\mathbf{c} \in \mathbb{R}^n$ and $s \geq 2\omega(\sqrt{\log n})$, and for every $\mathbf{c} \in \Lambda$, the min-entropy of $\mathcal{D}_{\Lambda, s, \mathbf{c}}$ is at least $n - 1$.*

We include some lemmas that describe the distributions of some lattice operations.

Lemma 2 (Adapted from [GPV08, Lemma 2.9]). *For any n -dimensional lattice Λ , $\mathbf{c} \in \Lambda$, and $s = \omega(\sqrt{\log n})$,*

$$\Pr_{x \leftarrow \mathcal{D}_{\Lambda, s, \mathbf{c}}} [\|\mathbf{x} - \mathbf{c}\|_2 > s\sqrt{n}] = \text{negl}(n).$$

Lemma 3 (Leftover Hash Lemma [ILL89, BS23]). *Let \mathbf{H} be a $(1 + \nu)/N$ -universal hash function family over (K, S, T) , where $N := |T|$. That is, for any distinct $x, x' \in S$,*

$$\Pr_{k \leftarrow \mathbb{S}K} [\mathbf{H}(k, x) = \mathbf{H}(k, x')] \leq (1 + \nu)/N.$$

If k, x_1, \dots, x_ℓ are mutually independent random variables, where k is uniform over K and each x_i is over S and has min-entropy at least h , then the statistical distance between $(k, \mathbf{H}(k, x_1), \dots, \mathbf{H}(k, x_\ell))$ and the uniform distribution on $K \times T^\ell$ is at most $\ell\sqrt{2^{-h}N} + \nu/2$.

Lemma 4 (Adapted from [Pei10, Theorem 3.1]). *Let $s_1, s_2 > 0$ with $s^2 = s_1^2 + s_2^2$. Let Λ_1, Λ_2 be n -dimensional lattices and $s_1, s_2 = \omega(\sqrt{\log n})$, and let $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^n$ be arbitrary. Consider the following probabilistic experiment:*

Choose $\mathbf{x}_2 \leftarrow \mathcal{D}_{\Lambda_2, s_2, \mathbf{c}_2}$ then choose $\mathbf{x}_1 \leftarrow \mathbf{x}_2 + \mathcal{D}_{\Lambda_1, s_1, \mathbf{c}_1 - \mathbf{x}_2}$.

The marginal distribution of \mathbf{x}_1 is statistically indistinguishable from $\mathcal{D}_{\Lambda_1, s, \mathbf{c}_1}$.

2.7 Canetti-Krawczyk model

We will prove LATKE’s IBKA secure in (a modified version of) the Canetti-Krawczyk (CK) model [CK01]. This is a standard model for key agreement protocols, and the same model that CHIP’s choice of IBKA [FG10] uses.

Similarly to [FG10], we do not include the property of forward secrecy,¹ and we do include the property of key compromise impersonation (KCI) resistance (formalized in [Kra05]). In words, KCI resistance ensures that the scenario wherein Mallory compromises Alice, then impersonates Bob to Alice, is infeasible. In a KCI-resistant protocol, the only person Mallory could impersonate is Alice herself. We make additional, nonstandard modifications, and argue their appropriateness in Section 3.3.1.

Model definitions. The authors of [CK01] present a game-based approach to modeling adaptive adversaries in an authenticated key agreement protocol. The adversary is permitted to start new sessions between honest parties, intercept their messages, inject messages, and corrupt them in various ways. To keep track of which parties are honest, we use the terminology of *owner* of the session—the party at which the session exists, i.e., the party for which a session state is being stored by the simulator (or an honest party, in the real world). The owner can be either the initiator or responder of the protocol. We may assume every session has an owner (otherwise, the session is just the adversary talking to itself). The party which whom an owner wishes to establish a shared secret is called a *peer*. A peer may also be controlled by the simulator (or an honest party in the real world).

Formally, the CK model permits the adversary \mathcal{A} to query the following oracles regarding the state of protocol sessions and participating parties. We use tr as shorthand for $(\text{tr}_{\text{out}}, \text{tr}_{\text{in}})$ and tr^T for $(\text{tr}_{\text{in}}, \text{tr}_{\text{out}})$.

NewSession($\text{id}_A, \text{id}_B, \text{role}$) Creates a new session where id_A is the owner and id_B is the peer, and specifies the *role* (**initiator** or **responder**) of id_A in the session. If **initiator**, then id_A may send a message to id_B as the protocol specifies.

Send($\text{id}_A, \text{id}_B, m$) Sends a message m from id_B to id_A , where id_A is the owner of an existing session.

Corrupt(id_A) Reveals to \mathcal{A} all the long-term keys belonging to party id_A .

RevealKey($\text{id}_A, \text{id}_B, \text{tr}$) Reveals to \mathcal{A} the shared secret derived by party id_A in the given *completed* session.

RevealState($\text{id}_A, \text{id}_B, \text{tr}$) Reveals to \mathcal{A} the session state of party id_A in the given *incomplete* session.

Test($\text{id}_A, \text{id}_B, \text{tr}$) Flips a coin b . If $b = 0$, reveals to \mathcal{A} the owner’s copy of the shared secret in the given *completed* session.² If $b = 1$, sends to \mathcal{A} a uniformly selected value from the shared secret space.

The following security definition measures an adversary’s ability to tell the difference between the real shared secret of the **Test** session and a uniformly chosen one. The definition requires the notion of *cleanness* in order to prevent trivial attacks:

¹The authors of [FG10] show *weak forward secrecy* of their IBKA, but this property is not used by CHIP. See Section 3.3.1.

²This is made slightly more specific than the original definition, which assumes that both parties have the same shared secret. This will not be the case in our construction. Revealing the owner’s secret key is the appropriate choice, since this is the party that the adversary hasn’t necessarily corrupted in some way.

Definition (Session cleanness). *A session in the CK model is clean iff the adversary did not have access to the session’s state upon creation. This means the adversary did not call `RevealState` during its establishment, or `RevealKey` upon completion on (id_A, id_B, tr) or (id_B, id_A, tr^T) . The adversary may have called `Corrupt`(id_A) or `Corrupt`(id_B), but if it did, it was not actively controlling the party during the session’s establishment.*

Definition (Session key (SK) security with KCI resistance). *Let Π be a key agreement protocol, and let \mathcal{A} be an adversary. Consider the following security experiment. \mathcal{A} is permitted to arbitrarily interact with the oracles defined above, on the condition that it makes at most one `Test` query, whose contents we denote by (id_A, id_B, tr) . Eventually \mathcal{A} terminates and outputs its single-bit response b' to the test query.*

Π is SK-secure with KCI resistance iff the advantage of any PPT \mathcal{A} at guessing b is negligible, where (id_A, id_B, tr) is clean, (id_B, id_A, tr^T) is clean if it exists, and `Corrupt` has never been called on the session peer. That is, with these conditions,

$$\text{Adv}_{\mathcal{A}}^{\text{sk}} := |\Pr[b = b'] - 1/2| = \text{negl}(n).$$

2.8 Universal Composability

The Universal Composability (UC) model, introduced by Canetti [Can01], is an alternative to the game-based model of security more commonly used in cryptography. Broadly, the model frames cryptographic protocols as idealized *functionalities*, which can be thought of as black boxes with a tightly constrained interface to the outside world. The UC simulation-based security definition guarantees that these functionalities can be concurrently composed with each other and used as subprotocols in larger protocols. Informally, the main security definition is that a protocol Π *UC-realizes* a functionality \mathcal{F} iff there exists a simulator with access to \mathcal{F} that can convincingly simulate an execution of Π to an adversary who believes it can communicate with all the parties of Π . A protocol Π *UC-realizes* \mathcal{F} *in the \mathcal{F}' -hybrid model* iff it realizes \mathcal{F} when all parties in Π and the adversary are permitted to make queries to an unbounded number of copies of \mathcal{F}' . The UC composition theorem states that, if Π' UC-realizes \mathcal{F}' , then Π using Π' UC-realizes \mathcal{F} in the bare model, i.e., we can compose protocols that realize functionalities, and they behave as if they were the composition of the functionalities themselves.

iPAKE functionality. The CHIP framework we build on is proven secure in the UC framework assuming the existence of a UC-secure PAKE. That is, it UC-realizes the $\mathcal{F}_{\text{iPAKE}}$ functionality in the $\mathcal{F}_{\text{PAKE}}$ model. Since our construction does not require a re-evaluation of the security proofs of CHIP, we omit the definitions of both these functionalities. Full definitions can be found in [CNPR22].

The only details about the $\mathcal{F}_{\text{iPAKE}}$ we will need to keep track of for our use are the session identifiers. Each new session in $\mathcal{F}_{\text{iPAKE}}$ is associated with a *static session identifier* sid , which identifies a group of parties who wish to authenticate amongst themselves, e.g., devices in a particular room in a building, $sid = \text{“Room 305”}$. In addition, prior to the beginning of a new iPAKE session, a *sub-session identifier* $ssid$ is established between

parties. This uniquely identifies that particular execution of the protocol. This can be established by out-of-band means, or by, e.g., exchanging nonces [BLR04]. For clarity of presentation, we will assume in our protocol definitions that the `sid` and `ssid` establishment has already occurred.

3 A lattice-based output-fuzzy IBKA

In this section we construct a lattice-based output-fuzzy identity-based key agreement protocol with the two properties that the CHIP construction requires, namely KCI resistance and keygen coin independence. The protocol is described in its entirety in Figure 4.

A starting point. We begin with the identity-based encryption (IBE) scheme presented in [GPV08]. This is shown in Figure 2, modified to be an IBKEM rather than an IBE. It is built using a linear preimage sampleable function scheme (`GenTrap`, `SampleDom`, `SamplePre`). To set up, a party calls `GenTrap` and derives a main keypair ($\text{msk} = \mathbf{T}, \text{mpk} = \mathbf{A}$). To derive a decryption key `usk` for identifier `id`, a party calls `SamplePre` and uses knowledge of the trapdoor information `msk` to compute a short solution \mathbf{E} such that $H(\text{id}) = \mathbf{A}\mathbf{E}$. To encapsulate, a party uses the main public key \mathbf{A} and the identifier string to encrypt a random plaintext. To decapsulate, a party uses their knowledge of `usk` to decrypt the ciphertext.

We may use this IBKEM as a building block for an IBKA using the generic compiler introduced in [BCGP08, Protocol 1]. At a high level, this protocol, given any (IB)KEM, runs two rounds of (IB)KEM encapsulations, i.e., $A \rightarrow B$ and $B \rightarrow A$, and passes the results through a key derivation function (KDF). For CCA-secure (IB)KEMs, the authors show that this is a KCI-resistant key agreement protocol, and that it provides no forward secrecy (i.e., an adversary who learns both participants’ decapsulation keys can decrypt past sessions).

This security result says nothing of the protocol’s keygen coin independence, however. In fact, this isn’t a property that can generically hold. For example, suppose a CCA-secure IBKEM is modified to send `mpk` along with the encapsulated key. This modified scheme is clearly still CCA-secure, but it would trivially break the keygen coin independence property of the IBKA that used it, since `mpk` is (part of) the output of `IBKEM.KeyGen(1n; H(pw))`. To build a scheme with keygen coin independence, we must do so in a white-box way.

Our starting point for LATKE is this generic construction, instantiated using the *CPA-secure* IBKEM described in Figure 2. On its face, this scheme may not be secure due to the weak choice of IBKEM. Further, some of the modifications we make in order to achieve keygen coin independence will break correctness, and break security even further. We will solve the correctness issue by using a *fuzzy PAKE* in place of a PAKE in the CHIP construction, and we will solve the security issue by relaxing (with justification) the security model itself. At the end, the IBKA will inherit the KCI resistance of [BCGP08], and will provide no forward secrecy.

Public Parameters. LWE params n, q, χ, m . Preimage sampleable function scheme (GenTrap, SampleDom, SamplePre) with sampling domain $\mathcal{D}_{\mathbb{Z}^m, s}$. Plaintext size $\ell = \text{poly}(n)$. A random oracle $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{n \times \ell}$.

IBE.Setup($1^n, 1^m, q$) \rightarrow (msk, mpk)	IBE.Enc($\mathbf{A}, \text{id}, \mathbf{m} \in \{0, 1\}^\ell$) \rightarrow ($\mathbf{c}_1, \mathbf{c}_2$)
1 : $(\mathbf{T}, \mathbf{A}) \leftarrow \text{GenTrap}(1^n, 1^m, q)$	1 : $\mathbf{U} := H(\text{id})$
2 : return (\mathbf{T}, \mathbf{A})	2 : $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$
	3 : $\mathbf{x}_1, \mathbf{x}_2 \leftarrow \chi^{m+\ell}$
IBE.Extract(mpki, msk, id) \rightarrow usk	4 : $\mathbf{c}_1 := \mathbf{A}^T \mathbf{s} + \mathbf{x}_1$
1 : $\mathbf{U} := H(\text{id})$	5 : $\mathbf{c}_2 := \mathbf{U}^T \mathbf{s} + \mathbf{x}_2 + \lfloor q/2 \rfloor \cdot \mathbf{m}$
2 : usk \leftarrow SamplePre(msk, \mathbf{U})	6 : return ($\mathbf{c}_1, \mathbf{c}_2$) $\in \mathbb{Z}_q^m \times \mathbb{Z}_q^\ell$
3 : return usk	
	IBE.Dec(uski = $\mathbf{E}, (\mathbf{c}_1, \mathbf{c}_2)$) \rightarrow b
	1 : $\mathbf{m}' := \mathbf{c}_2 - \mathbf{E}^T \mathbf{c}_1$
	2 : return $\lfloor \mathbf{m}' \rfloor$

Figure 2: The identity-based encryption mechanism from [GPV08]. $\lfloor \mathbf{v} \rfloor$ denotes the element-wise rounding operation which, for element i , equals 0 if $v_i \in \{-\lfloor q/4 \rfloor, \dots, \lfloor q/4 \rfloor\}$, and 1 otherwise.

3.1 Ambiguity

As we will see, the current IBKA is not keygen coin independent. We begin by analyzing the *uniform ambiguity* [BLMG21] of the underlying IBKEM.

Ambiguity and keygen coin independence. A KEM is *uniform-ambiguous* iff an adversary cannot determine the recipient of a ciphertext even if given the secret key. In other words, a uniform plaintext (i.e., the KEM shared secret) is indistinguishable from a mauled plaintext that resulted from decrypting under the wrong key. As shown in [BLMG21], there exist uniform-ambiguous KEMs from the computational Diffie-Hellman assumption. We extend this definition to identity-based KEMs.

Definition (usk Uniform Ambiguity). *An identity-based key exchange mechanism IBKEM is usk-uniform-ambiguous iff an adversary cannot determine the recipient of a ciphertext even if given the decryption key usk of one of the possible recipients. Concretely, for all probabilistic polynomial time adversaries \mathcal{A} ,*

$$\left| \Pr[\mathcal{U}\text{-AMB}_{\text{usk}}(\text{IBKEM}, \mathcal{A}) = 1] - \frac{1}{2} \right| = \text{negl}(n)$$

with the $\mathcal{U}\text{-AMB}_{\text{usk}}$ game defined as in Figure 3.

$\mathcal{U}\text{-AMB}_{\text{usk}}(1^\lambda, \text{IBKEM}, \mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1))$
1 : $\forall i \in \{0, 1\} : (\text{msk}_i, \text{mpk}_i) \leftarrow \text{IBKEM.Setup}(1^\lambda)$
2 : $(\text{id}_0, \text{id}_1, \text{state}) \leftarrow \mathcal{A}_0(1^\lambda, \text{mpk}_0, \text{mpk}_1)$
3 : $\forall i \in \{0, 1\} : \text{usk}_i \leftarrow \text{IBKEM.Extract}(\text{mpk}_i, \text{msk}_i, \text{id}_i)$
4 : $b \leftarrow_{\$} \{0, 1\}$
5 : $(\text{ss}, \text{ct}) \leftarrow \text{IBKEM.Encap}(\text{mpk}_b, \text{id}_b)$
6 : $b' \leftarrow \mathcal{A}_1(\text{state}, \text{usk}_0, \text{usk}_1, \text{ct})$
7 : return $b = b'$

Figure 3: The \mathcal{D} -ambiguity game from [BLMG21], restated with respect to IBKEMs and adversaries with knowledge of usk .

Recall that keygen coin independence requires that the protocol’s transcript not detectably depend on msk , usk , or any other value transitively derived from the **KeyGen** process. In particular, this requires that an adversary with some usk cannot distinguish between a ciphertext decryptable by usk and a ciphertext not decryptable by usk . Thus, usk -uniform ambiguity is a necessary property for keygen coin independence. As we will see later, it is not sufficient, but this serves as an useful intermediate goal.

Barriers to instantiation. Unfortunately, to the authors’ knowledge, no uniform-ambiguous KEM (let alone IBKEM) has been constructed from lattice assumptions. To explain why, we consider some existing lattice- and code-based KEMs—including FrodoKEM [NAB⁺20], NTRUPrime [CDH⁺20], Classic McEliece [ABC⁺22], and Kyber [SAB⁺22]—and observe that every one has to round or otherwise decode some value in the decapsulation step. This appears inherent. In every scheme, the cleartext-to-plaintext encoding requires the encapsulator to pick messages which are near, but not equal to, known threshold values. For example, **FrodoPKE.Enc**³ encodes a cleartext bit b as $b \cdot \lfloor q/2 \rfloor + e$, where $e \leftarrow \chi$ is some small Gaussian noise. Given the correct decryption key, **FrodoPKE.Dec** will round off the decrypted scalars to 0 or $\lfloor q/2 \rfloor$ to retrieve the original cleartext bits.

Given the wrong decryption key, however, **FrodoPKE.Dec** will return mangled message, one whose distribution is far from the one produced by **FrodoPKE.Enc**.⁴ This is a high-quality oracle for determining the correct secret key. If an adversary decrypts a ciphertext and finds a value that is

³We refer to the FrodoKEM public-key encryption scheme (PKE) rather than the KEM, because **FrodoKEM.Decaps** has an explicit equality check that tells the decapsulator if the correct key was used. The CPA-secure PKE thus has a better chance of being uniform-ambiguous. This applies to any KEM which results from the Fujisaki-Okamoto transform [FO99].

⁴Indeed, if **FrodoPKE.Dec** gets an sk whose difference from the true secret is Δ , it will return a plaintext of $\text{Frodo.Encode}(\mu) + \mathbf{E}''' - (\mathbf{S}'\mathbf{A} + \mathbf{E}')\Delta$, where \mathbf{E}''' is short. Thus, if the distribution of sk is not close to the true secret, the resulting plaintext distribution is much closer to uniform than Gaussian.

far from both 0 and $\lfloor q/2 \rfloor$, then it is overwhelmingly likely that they used the wrong decryption key.

Modification #1: Breaking correctness. The observation above yields an idea of how to adapt our IBKEM to be uniform-ambiguous: simply remove the cleartext-to-plaintext encoding step in `Encap`. Rather than encoding cleartext bits, an honest encapsulator may instead encrypt a uniform plaintext vector. This badly breaks correctness: the decapsulator in this modified scheme always recovers a secret that is some small distance away from the encapsulator’s secret.

To compound the correctness issue, it is unsafe to use common key reconciliation mechanisms, such as fuzzy extractors [DRS04], information reconciliation mechanisms [Pei14, RW04], or other coding theory techniques.⁵ This is because these schemes are built for parties who approximately know the same high-entropy secret. But the IBKA’s lack of forward secrecy means that *the secrets it produces are not high-entropy*. Since any passive adversary can determine the secrets by brute forcing `pw`, the shared secret distribution has at best the same entropy as the password distribution.

Modification #2: Breaking security. The noticeable correctness error necessitates another change in the IBKA: we can no longer pass the KEM shared secrets through a key derivation function (KDF) or hash. These functions, by definition, amplify small differences in their inputs. If the parties in a non-correct protocol input their shared secrets to a KDF or hash, they will receive two distinct and unrelated final secrets. So instead, the parties will simply sum the two shared secrets as elements of \mathbb{Z}_q^ℓ . As we will see later, this modification makes our IBKA trivially insecure in the CK model. Rather than patching it up to meet the definition of SK-security we will instead degrade the security definition to fit the IBKA in Section 3.3.1. We will see that the security of CHIP, and LATKE consequently, *does not require the stronger notion of SK security*.

Modification #3: Fixing correctness later. In order for parties in the LATKE iPAKE to derive the same final secret, it needs to solve the correctness issue at some point. Recent work [DHP⁺18] defines an instantiation a family of algorithms, called *fuzzy PAKEs* (fPAKEs), which solve precisely the issue of reconciling approximately equal low-entropy secrets. We will use a specific fPAKE instantiation as a drop-in replacement for the PAKE in the CHIP framework. This setup is quite convenient: CHIP requires the use of a PAKE regardless, so using a special PAKE that additionally does reconciliation spares LATKE the overhead of adding additional steps to the IBKA.

3.2 Instance shifting

Our IBKA is now `usk`-ambiguous, but this is not enough. Recall that a password-guessing adversary can use `pw` to derive `msk` and `mpk`. This is a problem for the current scheme. As shown in [MP12], an adversary with a trapdoor `msk` for `mpk = A` can efficiently invert LWE instances over `A`. Since the inversion algorithm is overwhelmingly likely to fail if the wrong

⁵A description of how the fuzzy extractor technique breaks down can be found in [DHP⁺18, Appendix H].

\mathbf{A} (and trapdoor) is used, this provides a high-quality password guessing oracle to any passive attacker.

Thus, it is necessary *shift* the existing setup to a new instance $\bar{\mathbf{A}}$ that is not trapdoored. This is a difficult needle to thread—parties cannot encrypt *any* messages using \mathbf{A} , but at the same time they need the identity property $\mathbf{A}\mathbf{E} = \text{H}_1(\text{id})$ in order to keep authenticity guarantees.

Modification #4: Encrypting in a non-trapdoored lattice. We resolve this by first introducing a public, uniform, unrelated matrix \mathbf{B} and, in the online phase of the IBKA, forcing each participant to provide a *public key* \mathbf{V} and prove knowledge of a short \mathbf{E}' such that $\mathbf{B}\mathbf{E}' = \mathbf{V}$. Since no ID is tied to \mathbf{B} , encrypting using just \mathbf{B} offers no authenticity. And encrypting using both \mathbf{A} and \mathbf{B} will still break passive security, since \mathbf{A} is trapdoored. Our solution is to encrypt using $\bar{\mathbf{A}} := \mathbf{A} + \mathbf{B}$ and public key $\text{H}_1(\text{id}) + \mathbf{V}$. Notice that the only obvious way to decrypt such a ciphertext is to have $\mathbf{E} = \mathbf{E}'$. We prove that indeed this must be the case. More specifically, we prove in Section 3.3.1 that, to authenticate $\text{H}_1(\text{id})$ wrt \mathbf{A} , it suffices to prove knowledge of \mathbf{E}' and authenticate $\text{H}_1(\text{id}) + \mathbf{B}\mathbf{E}'$ wrt $\mathbf{A} + \mathbf{B}$.⁶

Formally, we introduce an *authentication phase* of the iPAKE, wherein both parties present their public key \mathbf{V} along with a NIZK proof π of the *inhomogeneous short integer solution* (ISIS) relation:

$$R_{\text{ISIS}} = \{(\beta, \mathbf{B}, \mathbf{V}; \mathbf{E}) : \bigwedge_i \mathbf{B}\mathbf{e}_i = \mathbf{v}_i \wedge \|\mathbf{e}_i\|_2 \leq \beta\},$$

where $\mathbf{e}_i, \mathbf{v}_i$ refer to the columns of \mathbf{E}, \mathbf{V} , respectively. We will define β to be sufficiently small to prevent an authenticity attack, but sufficiently large so as not to break (fuzzy) correctness. Note, since \mathbf{V} does not change across executions, the authentication phase only needs to be run once per pair of participants. This is the final modification we need. The LATKE IBKA is described in its entirety in Figure 4.

In the following sections, we prove security, (fuzzy) correctness, and keygen coin independence of the IBKA. We then analyze the IBKA's efficiency, and move on to show how it can be composed securely with an fPAKE to produce a scheme that UC-realizes $\mathcal{F}_{\text{iPAKE}}$.

3.3 Proofs

We now show correctness and bound the decryption error of our IBKA. This comes directly from the correctness proof of the dual encryption scheme of [GPV08].⁷

Theorem 5 (Output-fuzziness). *Let (GenTrap, SampleDom, SamplePre) be a linear PSF from \mathbb{Z}^m to \mathbb{Z}_q^n with domain distribution $\mathcal{D}_{\mathbb{Z}^m, s}$, where $q \geq 20sm$. Let $\alpha \leq 1/(s\sqrt{m} \cdot \omega(\sqrt{\log n})) \leq 1/20$, and let $\chi = \mathcal{D}_{\mathbb{Z}, \alpha q}$. Then the IBKA described in Figure 4 is output-fuzzy with ℓ_∞ -distance $q/10$. That*

⁶Indeed, the proof of knowledge is necessary for security. Without it, a malicious party could pick a fresh \mathbf{F} and send $\mathbf{V} = \bar{\mathbf{A}}\mathbf{F} - \text{H}_1(\text{id})$ so that the new public key is $\text{H}_1(\text{id}) + \mathbf{V} = \bar{\mathbf{A}}\mathbf{F}$. This breaks authenticity.

⁷The fuzziness bound of $q/10$ bound is high, and somewhat arbitrary. It can be tweaked by choosing smaller α or larger q .

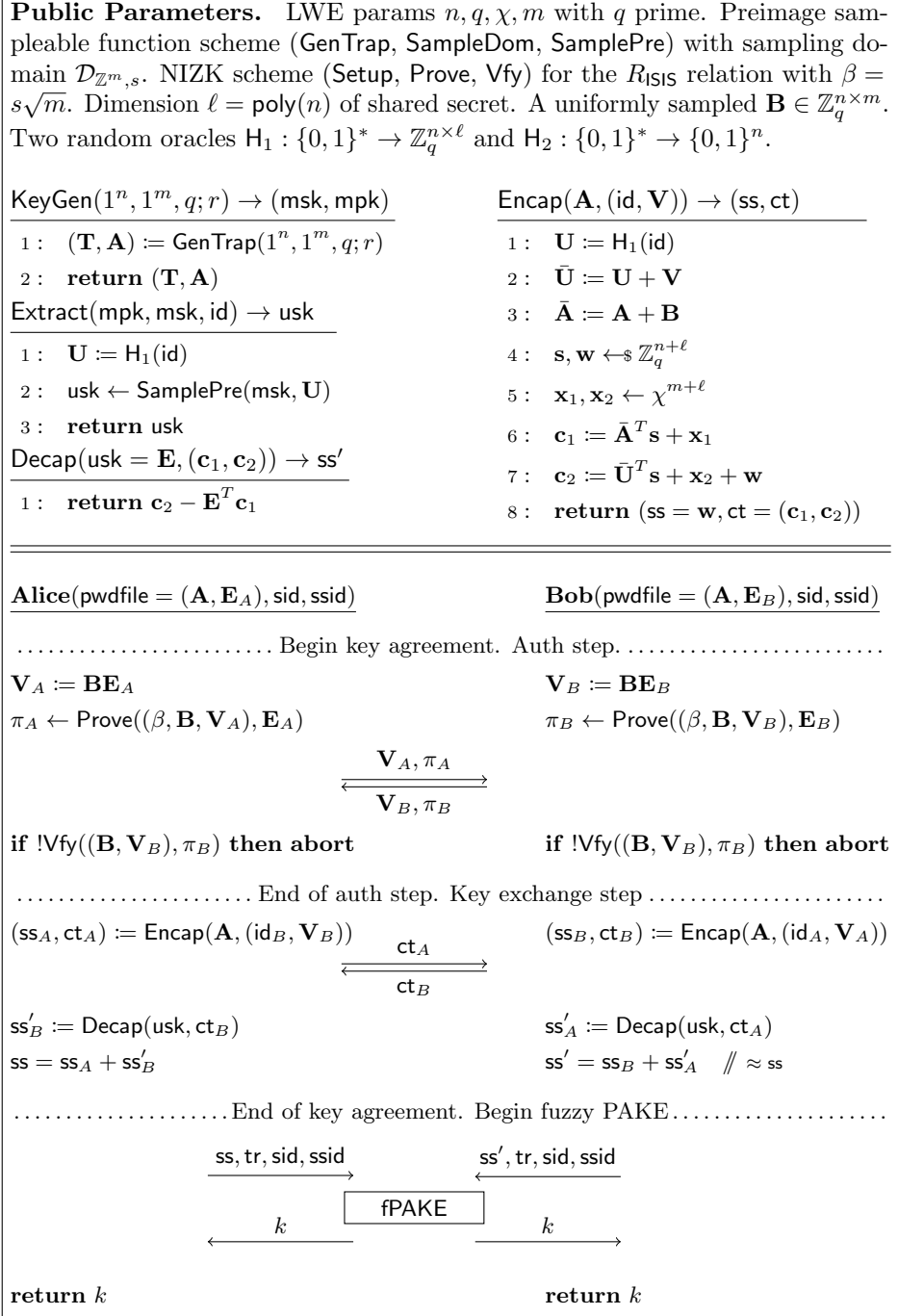


Figure 4: The LATKE iPAKE. GenPwfile is identical to CHIP.GenPwfile in Figure 1, using $\text{H} := \text{H}_2$. tr represents the protocol transcript up to that point. sid and ssid are session identifiers, as defined in Section 2.8. The NIZK setup is omitted, as this can be done at any time.

is, for any honestly generated msk , mpk , \mathbf{V} , and usk with corresponding identifier id ,

$$\Pr \left[\left\| \text{ss} - \text{ss}' \right\|_{\infty} > q/10 \mid \begin{array}{l} (\text{ss}_A, \text{ct}_A) \leftarrow \text{Encap}(\mathbf{A}, (\text{id}_A, \mathbf{V}_A)) \\ (\text{ss}_B, \text{ct}_B) \leftarrow \text{Encap}(\mathbf{A}, (\text{id}_B, \mathbf{V}_B)) \\ \text{ss}'_B := \text{Decap}(\mathbf{E}_A, \text{ct}_B) \\ \text{ss}'_A := \text{Decap}(\mathbf{E}_B, \text{ct}_A) \\ \text{ss} = \text{ss}_A + \text{ss}'_B \\ \text{ss}' = \text{ss}'_A + \text{ss}_B \end{array} \right] = \text{negl}(n).$$

Proof. Let $(\mathbf{c}_1, \mathbf{c}_2) = \text{ct}_A$. Then

$$\text{ss}'_A = \mathbf{c}_2 - \mathbf{E}_B^T \mathbf{c}_1 = \bar{\mathbf{U}}_B^T \mathbf{s} + \mathbf{x}_2 + \text{ss}_A - \mathbf{E}_B^T (\bar{\mathbf{A}}^T \mathbf{s} + \mathbf{x}_1),$$

where $\mathbf{x}_1, \mathbf{x}_2 \leftarrow \chi^{m+\ell}$ and $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. Since \mathbf{E}_B is such that $\bar{\mathbf{A}} \mathbf{E}_B = \bar{\mathbf{U}}_B$,

$$\text{ss}'_A = \bar{\mathbf{U}}_B^T \mathbf{s} + \mathbf{x}_2 + \text{ss}_A - \bar{\mathbf{U}}_B^T \mathbf{s} - \mathbf{E}_B^T \mathbf{x}_1 = \text{ss}_A + \mathbf{x}_2 - \mathbf{E}_B^T \mathbf{x}_1.$$

Note the distribution from which \mathbf{E}_B is extracted is statistically indistinguishable from the conditional distribution $\mathcal{D}_{\mathbb{Z}^m \times \ell, s}$ given $\mathbf{A} \mathbf{E}_B = \mathbf{U}_B$. Let $\mathbf{h} = \mathbf{E}_B^T \mathbf{x}_1 \in \mathbb{Z}_q^\ell$. Following the Cauchy-Schwartz argument in [GPV08, Lemma 8.2], we bound each $|h_i| < q/20$ with overwhelming probability. Since $\mathbf{x}_2 \sim \mathcal{D}_{\mathbb{Z}^\ell, \alpha q}$, applying Lemma 2 to each coordinate yields $\|\mathbf{x}_2\|_{\infty} \leq \alpha q \leq q/20$ with overwhelming probability as well. Thus $\|\text{ss}_A - \text{ss}'_A\|_{\infty} \leq q/10$ with overwhelming probability. Since $\text{ss} - \text{ss}' = (\text{ss}_A - \text{ss}'_A) + (\text{ss}_B - \text{ss}'_B)$, we double the bound.

Completing the correctness claim, we note that by Lemma 2, the choice of $\beta = s\sqrt{m}$ in Figure 4 is large enough that an honestly generated $\mathbf{E} \leftarrow \text{SamplePre}(\text{H}_1(\text{id}))$ satisfies R_{SIS} with overwhelming probability. Thus, honest proofs verify successfully. \square

We now prove keygen coin independence of our IBKA. There are two subtleties here. First, the reason the authentication step doesn't leak any information about msk is that **Extract** procedure is non-deterministic and has high-entropy. This, and the randomness of \mathbf{B} , is sufficient to hide msk . Second, instance shifting is what allows us to show that the ciphertexts are indistinguishable from randomness under the LWE assumption. Namely, because \mathbf{B} is a public parameter and perfectly hides \mathbf{A} , we can replace it with a (function of an) LWE challenge matrix.

Theorem 6 (Keygen coin independence). *Let q be prime. Let $(\text{GenTrap}, \text{SampleDom}, \text{SamplePre})$ be a linear PSF from \mathbb{Z}^m to \mathbb{Z}_q^n with domain distribution $\mathcal{D}_{\mathbb{Z}^m, s}$, SamplePre min-entropy at least $m-1$, and $m \geq 2n \log q$. Let $\alpha > 0$ be such that $\text{LWE}_{q, n, m, \chi}$ is hard, where $\chi := \mathcal{D}_{\mathbb{Z}, \alpha q}$. Let $(\text{Setup}, \text{Prove}, \text{Vfy}, \text{Ext}, \text{Sim})$ be a NIZK for R_{SIS} . Then the IBKA described in Figure 4 is keygen coin independent.*

Proof. We will use a sequence of hybrid distributions of an adversary's view, ending in a view whose values are independent of the randomness used in **KeyGen**. For clarity, we will only consider a single transcript, but the argument applies identically if you perform the same transformation on all $\text{poly}(n)$ -many transcripts in each hybrid.

Hybrid 0. The adversary’s view of a protocol execution (including public parameters) is $(\mathbf{B}, \mathbf{V}_A, \mathbf{V}_B, \pi_A, \pi_B, \text{ct}_A, \text{ct}_B)$, where each value is honestly computed.

Hybrid 1. Replace \mathbf{V}_A and \mathbf{V}_B with uniformly selected vectors. We argue that this is statistically indistinguishable from the previous hybrid. We will show this for \mathbf{V}_A . An identical argument proves this for \mathbf{V}_B .

Recall $\mathbf{V}_A = \mathbf{B}\mathbf{E}_A$. \mathbf{E}_A is *not* independent of the keygen’s random coins: given challenge r_b , an adversary can generate $(\cdot, \hat{\mathbf{A}}) := \text{GenTrap}(1^n, 1^m, q; r_b)$. Then $\hat{\mathbf{A}}\mathbf{E}_A = \mathbf{H}_1(\text{id}_A)$ iff $b = 0$ with overwhelming probability. We must argue, then, that \mathbf{E}_A does not leak in this protocol, i.e., multiplication by \mathbf{B} sufficiently hides \mathbf{E}_A .

Indeed, $\mathbf{B}\mathbf{E}_A$ is statistically close to uniform. By assumption, the `SamplePre` output distribution has min-entropy $h \geq m - 1$. Further, since q is prime, the choice of $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$ defines a ϵ -universal hash function family, where $\epsilon = q^{-n} = \text{negl}(n)$. By Lemma 3, $(\mathbf{B}, \mathbf{B}\mathbf{E}_A)$ is ϵ' -close to the uniform distribution on $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{n \times \ell}$, where $\epsilon' \leq \ell \sqrt{2^{-h} q^n} / 2$. Since $m \geq 2n \log q$, we have $\epsilon' = \text{negl}(n)$.

Hybrid 2. Replace the proofs π_A, π_B with ones generated by `Sim` on \mathbf{V}_A and \mathbf{V}_B , respectively. Since the NIZK is zero-knowledge by assumption, this view is indistinguishable from the last.

Hybrid 3. Replace \mathbf{c}_2 in both ciphertexts with uniformly random vectors. This view is perfectly indistinguishable from the last, since \mathbf{c}_2 is blinded by the uniform vector \mathbf{w} , and independent from the rest of the inputs.

Hybrid 4. Replace \mathbf{c}_1 in both ciphertexts with a uniformly random vectors. We show the reduction from $\text{LWE}_{q,n,m,\chi}$ for one the \mathbf{c}_1 . The argument for the other is identical.

Given an LWE challenge (\mathbf{C}, \mathbf{d}) , let $\bar{\mathbf{A}} := \mathbf{C}$, $\mathbf{B} := \bar{\mathbf{A}} - \mathbf{A}$, and $\mathbf{c}_1 := \mathbf{d}$. In the case the challenge is an LWE sample, $(\mathbf{B}, \mathbf{c}_1)$ is distributed as in the previous view. In the case the challenge is uniform, $(\mathbf{B}, \mathbf{c}_1)$ is distributed uniformly. □

3.3.1 Security in the CK model

CHIP’s IBKA [FG10] is proven secure in the Canetti-Krawczyk (CK) model, with the additional properties of KCI resistance and *weak forward secrecy* (wFS)—forward secrecy but only for sessions wherein the adversary was passive. We cannot prove our IBKA secure in the same model. The first reason is our IBKA is not weak forward-secret. This is a non-issue, though, since the CHIP construction explicitly composes with a PAKE in order to confer forward secrecy on an otherwise non-forward-secret IBKA. More concretely, the proof of security [CNPR22, Theorem 1] does not make use of wFS at any point.

If we use the non-wFS CK variant presented earlier in Section 2.7, our IBKA is still trivially insecure.⁸ The security model can be refined, though.

⁸ We break the SK-security with KCI resistance game defined in Section 2.7. Let $(\text{id}_A, \text{id}_B, \text{ct}_A, \text{ct}_B)$ be a clean, completed session where neither party is corrupted. We show that an attacker, Mallory, can derive the session’s shared secret. Let ss_A and ss_B denote the decapsulation of ct_A and ct_B , respectively. First Mallory corrupts some other party P

We make two observations: (1) `RevealKey` and `RevealState` appear to be the primary attack vectors against the IBKA, and (2) the proof of CHIP’s security [CNPR22, Theorem 1] *does not require that the IBKA be secure when the shared secret is revealed*. The second point is fairly intuitive upon inspection: if CHIP sends the output of the IBKA directly into the PAKE, and deletes all the intermediate state, there no reason to consider the direct leakage of the IBKA’s state or output in the threat model. The most an adversary can do to guess IBKA’s output is by performing an attack on the underlying PAKE, but its security implies that the adversary gets roughly 1 guess per attack. The argument against modeling reveal queries goes further: hiding the output and intermediate state of the IBKA is actually *essential* to CHIP’s security. If an adversary were allowed to see either, it would break keygen coin independence, and thus break the security of the entire iPAKE.⁹

nrSK-security. With this in mind, we define a restriction of the CK model from Section 2.7. We refer to the resulting security notion as *no-reveal session-key (nrSK) security with KCI resistance*. In this model, adversaries are not permitted to make `RevealKey` or `RevealState` queries. We note that our starting scheme [BCGP08] also does not model the revelation of decapsulated ciphertexts in `RevealState` for the same reasons as above. In fact, the same SK-security attack applies in the case that decapsulated keys aren’t omitted from the response. We now show that our IBKA is secure in this new model.

Theorem 7. *Let $(\text{GenTrap}, \text{SampleDom}, \text{SamplePre})$ be a linear PSF from \mathbb{Z}^m to \mathbb{Z}_q^n with domain distribution $\mathcal{D}_{\mathbb{Z}^m, s}$ and $s \geq \omega(\sqrt{\log n})$. Let $(\text{Setup}, \text{Prove}, \text{Vfy}, \text{Ext}, \text{Sim})$ be a NIZK for R_{ISIS} with $\beta := s\sqrt{m}$. Let $\alpha < 1/(s\sqrt{m} \cdot \omega(\sqrt{\log n}))$ and $\chi := \mathcal{D}_{\mathbb{Z}, \alpha q}$. Then the IBKA described in Figure 4 with these parameters is nrSK-secure with KCI resistance assuming the hardness of $\text{LWE}_{q, n, 2m+\ell, \chi'}$, where $\chi' := \mathcal{D}_{\mathbb{Z}, \alpha' q}$ and α' is such that $\sqrt{(\alpha'q)^2(1 + \beta^2) - (\alpha q)^2} = \omega(\sqrt{\log n})$.*

Proof. Suppose \mathcal{A} is an adversary against the nrSK-security game with noticeable advantage. \mathcal{C} will be the challenger, simulating a view to \mathcal{A} in the modified CK model. We will present a sequence of games which slightly

and initiates a session s with id_B using ciphertext ct_A . She receives some ciphertext ct and decrypts it using P ’s private key to get k . Finally, she calls `RevealKey` on the session and gets B ’s copy of the shared secret k' . Since $k' \approx k + \text{ss}_A$, Mallory can compute $\text{ss}_A \approx k' - k$. Using the same method, Mallory can derive (an approximation of) ss_B . Now that she knows the approximate sum $\text{ss}_A + \text{ss}_B$, Mallory can win `Test` with noticeable probability. Note, this is not fixed by summing a hash of context into k —it is a hash of public info, so she can simply be subtracted back out.

⁹We use the notation from [FG10] to describe the following brute force attack. Let the attacker, Mallory, have identifier id_M and public key be $r_M := g^{k_M}$ for a uniform $k_M \leftarrow \mathbb{Z}_q$. Mallory doesn’t know her `pwfile` $= (y, s_M)$. Let Mallory initiate an IBKA session with Alice, and then call `RevealKey` on the session, yielding the shared secret ss . Of the intermediate values in that session, Mallory knows z_2 , since it is readily computable without `pwfile`, and does not know z_1 . The only unknown variables in z_1 are y and s_M . Notice also that $y = g^x$ and $s_M = k_M + x\text{H}'_1(M, kr_M)$, so y and s_M are uniquely defined and efficiently computable once x is chosen. Thus, Mallory can brute force values of $x = \text{H}_2(\text{sid}, \text{pw})$ until the shared secret $\text{H}'_2(z_1, z_2)$ matches ss . In a similar attack, Mallory can use `RevealState` to dump a session’s z_1 before completion and brute force the password in the same way.

change \mathcal{A} 's view, but not enough that its advantage will go from noticeable to not-noticeable. The final game's view is independent of the challenge bit b . Since the final game admits no advantage for \mathcal{A} , we conclude that \mathcal{A} cannot distinguish $b = 0$ from $b = 1$ in the original game.

Recall in the definition of SK security that the adversary is permitted to have corrupted the owner of the session (i.e., the party that the adversary is not impersonating). We may assume wlog that this is the case. Further, since we do not need to prove forward secrecy, this will be the only case we have to consider. Let $Q \in \text{poly}(n)$ be a bound on the total number of participants that \mathcal{A} will create during the game.

Game 0. This is the base game. \mathcal{C} computes all the sessions itself, using `GenTrap` upon setup, and `Extract` to create the `usk` for new parties. The responses to `Test` are honest.

Game 1. \mathcal{C} designates a random $i^* \leftarrow_{\$} \{1, \dots, Q\}$ to be the test session. It will also designate a random party id_{P^*} (from $2Q$ possible peers, at most) to be in the peer in the test session. If \mathcal{A} doesn't choose i^* to be the test session, or id_{P^*} isn't the peer, then \mathcal{C} aborts.

Analysis of Game 1. If the \mathcal{A} 's probability of success in Game 0 is ϵ , the probability of success in Game 1 is $\epsilon/(2Q^2)$. Thus if ϵ is noticeable, \mathcal{A} 's performance here is also noticeable.

Game 2. We remove the need for a PSF trapdoor. At the beginning of the game, \mathcal{C} samples $\mathbf{A}, \mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$. For every hash query $H_1(\text{msg})$, \mathcal{C} will sample a $\mathbf{h} \leftarrow \mathcal{D}_{\mathbb{Z}^m, s}$, save (msg, \mathbf{h}) , and return $\mathbf{A}\mathbf{h}$. When creating a new party given by `id`, \mathcal{C} just sets `usk` := \mathbf{h} if one exists for that `id`. Otherwise, it will make a fresh query and use that \mathbf{h} .

Game 1 $\stackrel{s}{\approx}$ Game 2. By hypothesis, \mathbf{A} is statistically close to uniform, and a sample $\mathbf{h} \leftarrow \text{SamplePre}(\mathbf{t})$ is statistically close to the conditional distribution $\mathcal{D}_{\mathbb{Z}^m, s}$ given $\mathbf{A}\mathbf{h} = \mathbf{t}$. Thus, the distribution of `usk` values is not changed. Further, `SampleDom`, by definition of PSF, must produce \mathbf{h} such that $\mathbf{A}\mathbf{h}$ is (statistically close to) uniform.

Game 3. At the beginning of the game, \mathcal{C} replaces the test session peer's public key with a uniform $\mathbf{U}_{P^*} \leftarrow_{\$} \mathbb{Z}_q^n$, and programs $H_1(\text{id}_{P^*}) := \mathbf{U}_{P^*}$. Similarly, to simulate the party id_{P^*} , \mathcal{C} selects a uniform $\mathbf{V}_{P^*} \leftarrow_{\$} \mathbb{Z}_q^n$, and makes use of the efficient simulator $\text{Sim}(\beta, \mathbf{B}, \mathbf{V}_{P^*})$ to generate the proof π .

Game 2 $\stackrel{s}{\approx}$ Game 3. By the same argument as the previous hop, \mathbf{V} and \mathbf{U} statistically indistinguishable from uniform in both games. And since the peer is uncorrupted by assumption, it's never explicitly told a preimage to \mathbf{U} or \mathbf{V} . And by the zero-knowledge assumption, `Sim` produces proofs which are distributed identically to honestly generated proofs.

Note that \mathcal{C} can fully simulate the execution of every party except for id_{P^*} . In particular, since it no longer has its decryption key \mathbf{E}_{P^*} , it cannot compute id_{P^*} 's view of the shared secret. This doesn't affect the simulator's behavior, though. Recall `RevealKey` is not allowed, and \mathcal{C} only needs to return the *owner's* shared secret in `Test`. Since \mathcal{C} can compute this value normally, the test session is unaffected.

Game 4. In the test session, the challenger modifies the session owner's ciphertext ct_{O^*} . The \mathbf{c}_1 value is sampled uniformly. And $\mathbf{c}_2 := \mathbf{y} + \mathbf{w}$,

where $\mathbf{y}, \mathbf{w} \leftarrow \mathbb{Z}_q^\ell$ are both uniform. The challenger also decapsulates the peer's ciphertext ct_{P^*} to the key \mathbf{d} . The **Test** response is either $\text{ss}^{(0)} := \mathbf{w} + \mathbf{d}$ on $b = 0$, or a uniform value $\text{ss}^{(1)} \leftarrow \mathbb{Z}_q^\ell$.

Game 3 \approx **Game 4**. We provide a reduction from the $\text{LWE}_{q,n,2m+\ell,\chi'}$ problem using the parameters described above. Denote the LWE challenge as $((\mathbf{A}, \mathbf{B}, \mathbf{U}), (\mathbf{A}_1, \mathbf{B}_1, \mathbf{a}_2))$. Concretely, in the case this is from the LWE distribution, then

$$\mathbf{a}_1 = \mathbf{A}^T \mathbf{s} + \mathbf{x}_{a,1} \quad \mathbf{b}_1 = \mathbf{B}^T \mathbf{s} + \mathbf{x}_{b,1} \quad \mathbf{a}_2 = \mathbf{U}^T \mathbf{s} + \mathbf{x}_{a,2},$$

where $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ is the LWE secret and all the $\mathbf{x}_1, \mathbf{x}_2$ values are from χ' with the appropriate dimension.

At the very beginning of the game, \mathcal{C} sets the global \mathbf{A}, \mathbf{B} as given, and sets $\mathbf{U}_{P^*} := \mathbf{U}$. We will use the LWE challenge to construct the test session ciphertext ct_{O^*} intended for the peer. In the case it is an LWE instance, the ciphertext will be a valid ciphertext. In the case it is uniform, the ciphertext will be uniformly distributed. We will set \mathcal{C} 's answer to the LWE game to be \mathcal{A} 's response to the **Test** query. If \mathcal{A} behaves noticeably differently on the two types of ciphertext, \mathcal{C} will have noticeable advantage in the LWE game.

In the test session, \mathcal{A} sends its public key \mathbf{V}_{P^*} . \mathcal{C} uses the efficient extractor $\text{Ext}(\beta, \mathbf{B}, \mathbf{V}_{P^*}, \pi)$ to extract the secret key \mathbf{F} such that $\mathbf{V}_{P^*} = \mathbf{B}\mathbf{F}$ and the columns $\|\mathbf{f}_i\|_2 \leq \beta = s\sqrt{m}$. If this fails, the game ends and \mathcal{C} returns 0 (this cannot happen except with negligible probability by hypothesis, so we eliminate this case).

To construct ct_{O^*} , \mathcal{C} uses the given LWE instance that has shorter error, and adds error as needed in order to make the output distribution fit the ciphertext distribution. \mathcal{C} samples $\mathbf{w} \leftarrow \mathbb{Z}_q^\ell$ and compensating error terms $\mathbf{x}_1 \leftarrow \mathcal{D}_{\mathbb{Z}^m, t}$ and \mathbf{x}_2 with $x_{2,i} \leftarrow \mathcal{D}_{\mathbb{Z}, t'_i}$ (for t and $\{t'_i\}$ specified below), and lets

$$\begin{aligned} \mathbf{c}_1 &:= \mathbf{a}_1 + \mathbf{b}_1 + \mathbf{x}_1 = (\mathbf{A} + \mathbf{B})^T \mathbf{s} + (\mathbf{x}_{a,1} + \mathbf{x}_{b,1}) + \mathbf{x}_1 \\ \mathbf{c}_2 &:= \mathbf{a}_2 + \mathbf{F}^T \mathbf{B}_1 + \mathbf{x} + \mathbf{w} = (\mathbf{U} + \mathbf{V})^T \mathbf{s} + (\mathbf{x}_{a,2} + \mathbf{F}^T \mathbf{x}_{b,1}) + \mathbf{x}_2 + \mathbf{w}, \end{aligned}$$

where the RHS is what holds in the case the challenge is an LWE sample. In the case that the challenge is uniform, then \mathbf{c}_1 is uniformly distributed due to \mathbf{a}_1 , and \mathbf{c}_2 is uniformly distributed due to \mathbf{a}_2 . It remains to show that $\mathbf{c}_1, \mathbf{c}_2$ are indistinguishable from honest ciphertexts in the case the challenge is an LWE sample.

We argue \mathbf{c}_1 is distributed as an honest ciphertext. By Lemma 4, and since $\alpha'q = \omega(\sqrt{\log n})$, $\mathbf{x}_{a,1} + \mathbf{x}_{b,1} \sim \mathcal{D}_{\mathbb{Z}^m, \alpha'q\sqrt{2}}$. Thus, picking t such that $\sqrt{2(\alpha'q)^2 + t^2} = \alpha q$ makes it so $\mathbf{c}_1 \sim \mathcal{D}_{\mathbb{Z}^m, \alpha'q}$ as desired, so long as $t = \omega(\sqrt{\log n})$, which is true by assumption.

We argue \mathbf{c}_2 is distributed as an honest ciphertext with the same \mathbf{s} as \mathbf{c}_1 . Note \mathbf{F} is fixed, and the randomness in \mathbf{c}_2 only comes from $\mathbf{x}_{a,2}$ and $\mathbf{x}_{b,1}$. Let $\mathbf{g} := \mathbf{F}^T \mathbf{x}_{b,1}$ and denote the columns of \mathbf{F} by $\{\mathbf{f}_i\}$. Since $\mathbf{x}_{b,1} \sim \mathcal{D}_{\mathbb{Z}^m, \alpha'q}$, then each component $g_i \sim \mathcal{D}_{\mathbb{Z}, \alpha'q\|\mathbf{f}_i\|_2}$. By the same reasoning as above, summing with other error $\mathbf{h} := \mathbf{x}_{a,2} + \mathbf{g}$ yields $h_i \sim \mathcal{D}_{\mathbb{Z}, \alpha'q\sqrt{1+\|\mathbf{f}_i\|_2^2}}$. Finally, for all $i = 1, \dots, \ell$, we pick $t'_i = \omega(\sqrt{\log n})$ such that, for $x_{2,i} \sim \mathcal{D}_{\mathbb{Z}, t'_i}$, $h_i + x_{2,i}$ is distributed as $\mathcal{D}_{\mathbb{Z}, \alpha q}$. Since, by

assumption, the distance between α' and α is sufficiently large, these $\{t'_i\}$ exist. This completes the reduction.

Game 4 advantage is 0. Since \mathbf{w} is uniform and independent from the rest of the transcript (i.e., \mathbf{c}_1 and \mathbf{c}_2), $\mathbf{ss}^{(0)}$ is uniform and independent from the rest of the transcript. Also, $\mathbf{ss}^{(1)}$ is uniform and independent by construction. Thus, \mathcal{A} has no information about b , and its advantage is 0. \square

3.4 Efficiency

We briefly analyze the efficiency of the IBKA scheme. The public keys in the authentication phase are $n\ell \log q$ bits. The authors of [LNP22] provide benchmarks for the communication cost of a proof of knowledge of a (Module-)LWE secret in their NIZK scheme. At the 128-bit security level, the total size of a proof and commitment is 14.4KB. Since the benchmark uses structured lattices with ring dimension 128, one can expect a $128 \times$ increase if using unstructured LWE. We also note that the authentication phase is a one-time cost for any pair of parties. Once a pair successfully performs this step, they can save each others public keys and use them for future key agreement sessions.

The communication cost of the rest of the key agreement protocol is two ciphertexts, or $\log q(m + \ell)$ bits. The computation cost is relatively small: `Encap` requires a matrix multiplication of size nm and of size $n\ell$, and `Decap` requires a matrix multiplication of $m\ell$.

4 Composing with a fuzzy PAKE

With a suitable IBKA in hand, we follow the blueprint of CHIP to construct an iPAKE. In order to achieve forward secrecy, we must pass the IBKA shared secrets through a PAKE. And since the two parties do not perfectly agree on the output, the PAKE must be *fuzzy*, i.e., tolerant to small disagreements in the password. This functionality, called $\mathcal{F}_{\text{fPAKE}}$, is made precise and instantiated in two different ways in [DHP⁺18]. The functionality is exactly that of an explicitly authenticated PAKE functionality, with the condition that passwords are considered correct if they are within some threshold δ of the real password, using some distance function.

For LATKE, we use the generic fPAKE defined in [DHP⁺18, Section 3], which is a domain-optimized two-party computation (2PC) protocol using Yao’s garbled circuits [Yao86], and permits *any distance function* that can be represented as a boolean circuit. Per Theorem 5, we set the distance threshold to $\delta := q/10$.

It remains only to address the questions of whether it is safe to use the fPAKE in place of a PAKE, and which distance function and cryptographic assumptions can be used to instantiate it.

4.1 Security

The proof that CHIP UC-realizes $\mathcal{F}_{\text{fPAKE}}$ functionality [CNPR22, Theorem 1] is immediately generalizable. We make two small modifications: (1) we replace the value y derived `GenPwfile` with `msk` (the y value is indeed the `msk` in [FG10]); (2) we modify the simulator’s behavior on `TestPwd` queries to mark password guesses within distance δ as correct, just as `TestPwd` does in $\mathcal{F}_{\text{fPAKE}}$. The latter change does not affect the simulation: given an fPAKE password guess `ss`, the simulator still runs through every password passed to `H1`, or compromised password, and attempts to re-derive `ss' ≈ ss` from it. Making sure that the password guess, if legitimate, must have resulted from one of these two sources is addressed in [CNPR22, Lemma 1], which applies to any KCI-resistant IBKA.

Thus, LATKE UC-realizes $\mathcal{F}_{\text{fPAKE}}$ functionality using parameters from the intersection of the hypotheses of Theorems 5 to 7.

4.2 Instantiating the fPAKE

We address some details regarding how to fit the 2PC fPAKE into LATKE.

Distance function. We would like to use the ℓ_∞ distance function for the fPAKE, both because it is appropriate for the structure of `ss`, and because it has an efficient boolean circuit complexity of $O(\ell \log q)$. We cannot directly use ℓ_∞ , though. Recall that the input to the fPAKE is `(ss, tr, sid, ssid)`. If the two parties disagree on the transcript or session identifiers, even at a single bit, then the PAKE should not succeed. But if the ℓ_∞ distance is used for all the inputs, this may be violated. This is simple to fix: since the distance function can be any boolean circuit, we simply use the ℓ_∞ on the `ss` portion of the input, and the discrete distance $d(x, y) = \delta \cdot (x == y)$ on the rest of the input, where δ is the distance threshold. The final distance function is the sum of these two. Thus, any inputs whose transcripts disagree will automatically fail the PAKE.¹⁰ The final number of gates in this circuit is $O(m\ell \log q)$.

Cryptographic assumptions. The 2PC fPAKE construction requires UC-secure oblivious transfer (OT), and relies on a generic transform [BCL⁺11] which allows it to operate over unauthenticated channels. Both of these constructions are instantiable from lattice assumptions. There is a variety of UC-secure OT from LWE [BCV19, BBB⁺22, BC15, PVW08]. The generic transformation also relies on UC-secure OT, in addition to collision-resistant hash functions and non-committing encryption, all of which are achievable from lattice assumptions in the random oracle model.¹¹

¹⁰Another way around this issue is to not input `ctx := (tr, sid, ssid)` to the fPAKE at all. The purpose of these values is to bind the resulting key to the session context. To achieve this, it suffices to pass just `ss` into the fPAKE, and perform a key derivation function on the result using `ctx` as the context.

¹¹The theorem statements in [BCL⁺11] claim to require the existence of *enhanced trapdoor permutations*, of which there are currently no known LWE-based instantiations. However, these permutations are only used insofar as their existence implies the existence of non-committing encryption and UC-secure oblivious transfer, and commitments. Non-committing encryption is instantiable from LWE [CDMW09, YKXT20]. And UC commitments can be constructed in the OT-hybrid model [Kil88, GIKW14]. Thus, the assumption can be removed.

4.3 Efficiency

The 2PC fPAKE construction uses a bespoke variant of the *dual-execution* method for multiparty computation (MPC) [MF06, HKE12] in order to achieve malicious security without using a generic malicious-secure protocol. We estimate the communication and computation costs of this protocol, using the breakdowns given in [DHP⁺18]. Let $b = \ell \log q$ be the bitlength of \mathbf{ss} . The ℓ_∞ distance function is a magnitude comparator composed with a threshold gate, which has boolean circuit complexity of roughly $4b$ (not including XORs, due to the optimization in [KS08]). The 2PC fPAKE scheme requires 5 communication rounds and roughly b OT operations, $8b$ symmetric encryptions, $4b$ symmetric decryptions, $16b$ hashes, 5 signing operations, and 5 signature verification operations.

5 Future work

There are several aspects to this work that could be improved to make a faster, more efficient, and more secure iPAKE.

CRISP from lattices. Recall the CRISP protocol is a *strong* iPAKE (siPAKE), meaning that it is robust to precomputation attacks. In the case of CHIP (and hence LATKE), an adversary who precomputes a table of \mathbf{mpk} values from various password guesses can use it to speed up a brute force attack, should they ever retrieve an \mathbf{mpk} from a user device. The CRISP construction uses algebraic techniques which rely on bilinear pairings in order to locally rerandomize its \mathbf{pfile} without breaking authenticity. It is unclear how these techniques could transfer to a lattice setting.

Post-quantum security. The use of lattice assumptions does not immediately imply that a scheme is secure against quantum adversaries. To prove quantum resistance, it is necessary to demonstrate a *quantum reduction*, rather than a Turing reduction. Our use of the random oracle model (ROM) in the proof of Theorem 7 does not directly require adaptive programming (i.e., programming H using knowledge of historical H queries), nor rewinding, so a reduction in the quantum ROM (QROM) [BDF⁺11] may be possible. However, depending on the NIZK, the \mathbf{Ext} function may require rewinding, which is not permitted in the model. We leave as an open question whether this scheme, or schemes like it admit quantum reduction proofs.

Structured lattices. It is likely the case that all of the constructions described in this paper can be hoisted to the Module-LWE (MLWE) setting. This modification would likely improve both communication costs and computational complexity in the key agreement protocol. The only place that a property of unstructured lattices is used is in the leftover hash lemma, Lemma 3, but several plausible ring analogues exist [MKMS21, LW20, LPR13, SS11] Finally, the NIZK and fPAKE schemes we use either already are, or can reasonably be instantiated from MLWE, possibly yielding more efficiencies.

Looser NIZKs. The NIZK scheme we use for LATKE is one of the most efficient lattice-based ones to date. But there may be additional efficiencies

available if LATKE could use a NIZK with *relaxed soundness* [LN17] where the extractor extracts an \mathbf{E} such that $\mathbf{A}\mathbf{E} = c\mathbf{V}$, where $c > 1$ and the columns have length $\|\mathbf{e}_i\|_2 < \beta'$ for some $\beta' > \beta$. That is, its extraction is not exact, and it violates shortness requirements to some extent. Some signature schemes use relaxed soundness for unforgeability [KLS18, BCN18], though it's not clear that the techniques extend to our setting.

Hamming-distance fPAKE. As mentioned above, the 2PC fPAKE uses some relatively heavyweight cryptographic primitives in order to support unauthenticated channels. There is a second construction described in [DHP⁺18], one that's specialized to *binary* passwords which are close in Hamming distance. If we modify the LATKE IBKA to take the most significant bit of the outputs, it produces Hamming-close binary shared secrets. Indeed, the likelihood that the msb of one entry in \mathbf{ss} differs from \mathbf{ss}' is roughly

$$\int_0^q t \Pr[X > t/2] dt \leq \int_0^q t \cdot 2 \exp\left(-\frac{t^2}{8r^2}\right) dt = 8r^2 \left(1 - \exp\left(-\frac{q^2}{8r^2}\right)\right)$$

where $X \sim \mathcal{D}_{\mathbb{R},r}$ is an approximation of the noise term $\mathbf{ss} - \mathbf{ss}'$. In words, this is the probability, over all t and i , that \mathbf{ss}_i is within radius $t/2$ of $q/2$ (where the msb flips), times the probability that the noise term pushes \mathbf{ss}'_i across the $q/2$ boundary. In practice, this error can be small. Empirically, the two parties disagree on the msb of one component with probability 1.2% when using parameters from FrodoKEM-640.

This setup is not immediately usable though. The Hamming distance fPAKE is built from EUF-CMA-secure one-time signatures and a *labeled implicit-only PAKE* functionality, i.e., a PAKE functionality where (1) each party receives a distinct, uniform bitstring (the *label*) along with session key, and (2) an attacker does not learn whether their password guess was correct. To the authors' knowledge, no labeled implicit-only PAKE has been constructed from lattice assumptions. Recent work [SGJ23] shows that any CPA-secure, anonymous KEM (of which, Kyber, Saber, Frodo, Classic McEliece, and NTRU are examples) yields a UC-secure PAKE, EKE-KEM, in the $(\mathcal{F}_{\text{RO}}, \mathcal{F}_{\text{HC}})$ -hybrid model. While the construction of EKE-KEM can easily be modified to include labels, it cannot easily be modified to be implicit-only. The issue is familiar: this property requires that the adversary cannot learn when trial decryptions succeed, i.e., the KEM must be uniform-ambiguous. The lack of such KEMs from post-quantum assumptions is precisely the problem we began with.

There exist some avenues forward, though. There are a number of lattice-based PAKEs which likely have this implicit-only property, due to the use of a Diffie-Hellman-like construction [DAL⁺17, YGWX20a, YGWX20b]. While these schemes only proven secure in the BPR model [BPR00], and not the UC model, it appears likely that they, or slight variants thereof are UC secure. Their structure is close to that of SPAKE2 [AP05], which has been shown to realize a slight weakening of $\mathcal{F}_{\text{PAKE}}$ [ABB⁺20].

Moving away from UC. A theme in the efficiency issues appears to be the use of the UC framework. While UC provides strong security guarantees for (i)PAKEs compared to the BPR model [CHK⁺05], the overhead of a

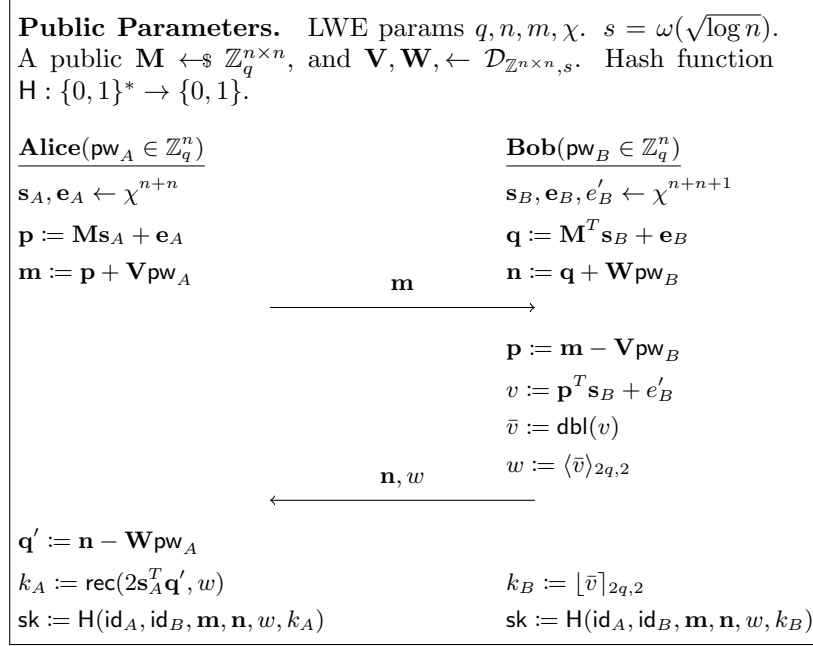


Figure 5: Proposal for an implicitly authenticated 1-bit fuzzy PAKE, modified from [BCNS15, DAL⁺17], in the model of SPAKE1 [AP05]. Key reconciliation notation rec , dbl , $\langle \cdot \rangle$, and $\lfloor \cdot \rfloor$ comes from [Pei14].

UC-secure fPAKE (or labeled implicit-only PAKE) is currently a barrier to real-world usability of LATKE. We outline a 1-bit LWE-based PAKE in Figure 5 which is naturally tolerant of short errors in the ℓ_p norm, for any p . It makes use of the key reconciliation mechanisms introduced by Peikert [Pei14]. Correctness comes from the fact that

$$\begin{aligned}
2\mathbf{s}_A^T \mathbf{q}' &= 2\mathbf{s}_A^T \mathbf{M}^T \mathbf{s}_B + 2\mathbf{s}_A^T \mathbf{e}_B - 2\mathbf{s}_A^T \mathbf{W}(\text{pw}_A - \text{pw}_B) \\
&\approx 2\mathbf{s}_A^T \mathbf{M}^T \mathbf{s}_B + 2\mathbf{e}_A^T \mathbf{s}_B + 2(\text{pw}_A - \text{pw}_B)^T \mathbf{V}^T \mathbf{s}_B + 2\mathbf{e}'_B = \text{dbl}(v)
\end{aligned}$$

when $\|\text{pw}_A - \text{pw}_B\|_p$ and the Gaussian parameter are sufficiently small.

The design of this PAKE mirrors that of SPAKE1 [AP05], and, similar to SPAKE1, it likely cannot be proven secure in the UC model. The barrier comes from the fact that the password is never input into the random oracle. Avoiding hashing makes sense for nonequal passwords—hashing would yield entirely unrelated outputs—but it frustrates the usual UC security arguments such as [AHH21, ABB⁺20].

This PAKE may have merits on its own: it is far more efficient than the fPAKE from 2PC, and it is plausibly provably secure in the BPR model from the Pairing With Errors (PWE) assumption [RG22]. Thus, it may be worthwhile to explore non-UC solutions to this problem.

6 Conclusion

We constructed and proved secure LATKE, an identity-binding balanced PAKE from lattice assumptions. To do so, we worked from the CHIP [CNPR22] model, replacing the underlying identity-based key agreement scheme (IBKA) with a new, less correct and less secure IBKA; and replacing the underlying PAKE with a fuzzy PAKE, which is tolerant to input errors. We scrutinized the security model and showed that using the new IBKA does not break overall security. We also showed that using the fuzzy PAKE resolves the correctness issues introduced by the new IBKA. Finally, we presented several directions for future work, including a potential replacement for our choice of fuzzy PAKE which uses less expensive cryptographic primitives, and a suggestion that this and other efficiency improvements may be incompatible with the very strong UC definition of iPAKE security.

7 Acknowledgements

Many thanks to Jonathan Katz for overall guidance and help in literature review. Thanks to Levi Schuck, Greg Rubin, Sanketh Menda, Doruk Gür, and Cas Cremers for providing feedback and answering questions.

References

- [ABB⁺20] Michel Abdalla, Manuel Barbosa, Tatiana Bradley, Stanislaw Jarecki, Jonathan Katz, and Jiayu Xu. Universally composable relaxed password authenticated key exchange. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 278–307. Springer, Heidelberg, August 2020.
- [ABC⁺22] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions>.
- [AHH21] Michel Abdalla, Björn Haase, and Julia Hesse. Security analysis of CPace. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 711–741. Springer, Heidelberg, December 2021.
- [AP05] Michel Abdalla and David Pointcheval. Simple password-based encrypted key exchange protocols. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 191–208. Springer, Heidelberg, February 2005.

- [BBB⁺22] Slim Bettaiieb, Loïc Bidoux, Olivier Blazy, Baptiste Cottier, and David Pointcheval. Post-quantum oblivious transfer from smooth projective hash functions with grey zone, 2022.
- [BBDQ18] Fabrice Benhamouda, Olivier Blazy, Léo Ducas, and Willy Quach. Hash proof systems over lattices revisited. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 644–674. Springer, Heidelberg, March 2018.
- [BC15] Olivier Blazy and Céline Chevalier. Generic construction of UC-secure oblivious transfer. In Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis, editors, *ACNS 15*, volume 9092 of *LNCS*, pages 65–86. Springer, Heidelberg, June 2015.
- [BCGP08] Colin Boyd, Yvonne Cliff, Juan González Nieto, and Kenneth G. Paterson. Efficient one-round key exchange in the standard model. In Yi Mu, Willy Susilo, and Jennifer Seberry, editors, *ACISP 08*, volume 5107 of *LNCS*, pages 69–83. Springer, Heidelberg, July 2008.
- [BCL⁺11] Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin. Secure computation without authentication. *Journal of Cryptology*, 24(4):720–760, October 2011.
- [BCN18] Cecilia Boschini, Jan Camenisch, and Gregory Neven. Relaxed lattice-based signatures with short zero-knowledge proofs. In Liqun Chen, Mark Manulis, and Steve Schneider, editors, *ISC 2018*, volume 11060 of *LNCS*, pages 3–22. Springer, Heidelberg, September 2018.
- [BCNS15] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *2015 IEEE Symposium on Security and Privacy*, pages 553–570. IEEE Computer Society Press, May 2015.
- [BCS16] Dan Boneh, Henry Corrigan-Gibbs, and Stuart E. Schechter. Balloon hashing: A memory-hard function providing provable protection against sequential attacks. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 220–248. Springer, Heidelberg, December 2016.
- [BCV19] Olivier Blazy, Céline Chevalier, and Quoc Huy Vu. Post-quantum UC-secure oblivious transfer in the standard model with adaptive corruptions. Cryptology ePrint Archive, Report 2019/707, 2019. <https://eprint.iacr.org/2019/707>.
- [BDF⁺11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011.

- [BDK15] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Argon2: the memory-hard function for password hashing and other applications. 2015.
- [BG03] Colin Boyd and Juan Manuel González Nieto. Round-optimal contributory conference key agreement. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 161–174. Springer, Heidelberg, January 2003.
- [BLMG21] Gabrielle Beck, Julia Len, Ian Miers, and Matthew Green. Fuzzy message detection. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 1507–1528. ACM Press, November 2021.
- [BLR04] Boaz Barak, Yehuda Lindell, and Tal Rabin. Protocol initialization for the framework of universal composability. Cryptology ePrint Archive, Report 2004/006, 2004. <https://eprint.iacr.org/2004/006>.
- [BM92] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *1992 IEEE Symposium on Security and Privacy*, pages 72–84. IEEE Computer Society Press, May 1992.
- [BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 139–155. Springer, Heidelberg, May 2000.
- [BS23] Dan Boneh and Victor Shoup. A graduate course in applied cryptography. Jan 2023.
- [Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.
- [CC04] Christian Cachin and Jan Camenisch, editors. *EUROCRYPT 2004*, volume 3027 of *LNCS*. Springer, Heidelberg, May 2004.
- [CDH⁺20] Cong Chen, Oussama Danba, Jeffrey Hoffstein, Andreas Hulsing, Joost Rijneveld, John M. Schanck, Peter Schwabe, William Whyte, Zhenfei Zhang, Tsunekazu Saito, Takashi Yamakawa, and Keita Xagawa. NTRU. Technical report, National Institute of Standards and Technology, 2020. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>.
- [CDMW09] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved non-committing encryption with applications to adaptively secure protocols. In Matsui [Mat09], pages 287–302.
- [CDVW12] Ran Canetti, Dana Dachman-Soled, Vinod Vaikuntanathan, and Hoeteck Wee. Efficient password authenticated key

- exchange via oblivious transfer. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 449–466. Springer, Heidelberg, May 2012.
- [CGIP12] Jean-Sébastien Coron, Aline Gouget, Thomas Icart, and Pascal Paillier. *Supplemental Access Control (PACE v2): Security Analysis of PACE Integrated Mapping*, pages 207–232. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
- [CHK⁺05] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 404–421. Springer, Heidelberg, May 2005.
- [CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Pfitzmann [Pfi01], pages 453–474.
- [CNPR22] Cas Cremers, Moni Naor, Shahar Paz, and Eyal Ronen. CHIP and CRISP: Protecting all parties against compromise through identity-binding PAKEs. In Dodis and Shrimpton [DS22], pages 668–698.
- [DAL⁺17] Jintai Ding, Saed Alsayigh, Jean Lancrenon, Saraswathy RV, and Michael Snook. Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In Helena Handschuh, editor, *CT-RSA 2017*, volume 10159 of *LNCS*, pages 183–204. Springer, Heidelberg, February 2017.
- [DHP⁺18] Pierre-Alain Dupont, Julia Hesse, David Pointcheval, Leonid Reyzin, and Sophia Yakoubov. Fuzzy password-authenticated key exchange. In Nielsen and Rijmen [NR18], pages 393–424.
- [DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Cachin and Camenisch [CC04], pages 523–540.
- [DS22] Yevgeniy Dodis and Thomas Shrimpton, editors. *CRYPTO 2022, Part II*, volume 13508 of *LNCS*. Springer, Heidelberg, August 2022.
- [FG10] Dario Fiore and Rosario Gennaro. *Identity-Based Key Exchange Protocols without Pairings*, pages 42–77. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
- [FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999.
- [GIKW14] Juan A. Garay, Yuval Ishai, Ranjit Kumaresan, and Hoeteck Wee. On the complexity of UC commitments. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*,

- volume 8441 of *LNCS*, pages 677–694. Springer, Heidelberg, May 2014.
- [GK10] Adam Groce and Jonathan Katz. A new framework for efficient password-based authenticated key exchange. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 516–525. ACM Press, October 2010.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.
- [Har15] Dan Harkins. Dragonfly Key Exchange. RFC 7664, November 2015.
- [HKE12] Yan Huang, Jonathan Katz, and David Evans. Quid-Pro-Quo-tocols: Strengthening semi-honest protocols with dual execution. In *2012 IEEE Symposium on Security and Privacy*, pages 272–284. IEEE Computer Society Press, May 2012.
- [Hv22] Feng Hao and Paul C. van Oorschot. SoK: Password-authenticated key exchange - theory, practice, standardization and real-world lessons. In Yuji Suga, Kouichi Sakurai, Xuhua Ding, and Kazue Sako, editors, *ASIACCS 22*, pages 697–711. ACM Press, May / June 2022.
- [ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24. ACM Press, May 1989.
- [JG04] Shaoquan Jiang and Guang Gong. Password based key exchange with mutual authentication. In Helena Handschuh and Anwar Hasan, editors, *SAC 2004*, volume 3357 of *LNCS*, pages 267–279. Springer, Heidelberg, August 2004.
- [JGH⁺20] Shaoquan Jiang, Guang Gong, Jingnan He, Khoa Nguyen, and Huaxiong Wang. PAKEs: New framework, new techniques and more efficient lattice-based constructions in the standard model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 396–427. Springer, Heidelberg, May 2020.
- [Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press, May 1988.
- [KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Nielsen and Rijmen [NR18], pages 552–586.
- [KOY01] Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient password-authenticated key exchange using human-memorable passwords. In Pfitzmann [Pfi01], pages 475–494.

- [Kra05] Hugo Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 546–566. Springer, Heidelberg, August 2005.
- [KS08] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 486–498. Springer, Heidelberg, July 2008.
- [KV09] Jonathan Katz and Vinod Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In Matsui [Mat09], pages 636–652.
- [LN17] Vadim Lyubashevsky and Gregory Neven. One-shot verifiable encryption from lattices. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 293–323. Springer, Heidelberg, April / May 2017.
- [LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In Dodis and Shrimpton [DS22], pages 71–101.
- [LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54. Springer, Heidelberg, May 2013.
- [LW20] Feng-Hao Liu and Zhedong Wang. Rounding in the rings. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 296–326. Springer, Heidelberg, August 2020.
- [Mat09] Mitsuru Matsui, editor. *ASIACRYPT 2009*, volume 5912 of *LNCS*. Springer, Heidelberg, December 2009.
- [MF06] Payman Mohassel and Matthew Franklin. Efficiency tradeoffs for malicious two-party computation. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 458–473. Springer, Heidelberg, April 2006.
- [MKMS21] Jose Maria Bermudo Mera, Angshuman Karmakar, Tilen Marc, and Azam Soleimani. Efficient lattice-based inner-product functional encryption. Cryptology ePrint Archive, Report 2021/046, 2021. <https://eprint.iacr.org/2021/046>.
- [MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

- [NAB⁺20] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM. Technical report, National Institute of Standards and Technology, 2020. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>.
- [NR18] Jesper Buus Nielsen and Vincent Rijmen, editors. *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*. Springer, Heidelberg, April / May 2018.
- [Pau22] Sebastian Paul. On the Transition to Post-Quantum Cryptography in the Industrial Internet of Things. May 2022.
- [Pei10] Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 80–97. Springer, Heidelberg, August 2010.
- [Pei14] Chris Peikert. Lattice cryptography for the internet. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 197–219. Springer, Heidelberg, October 2014.
- [Per09] Colin Percival. Stronger key derivation via sequential memory-hard functions, 2009. <http://www.tarsnap.com/scrypt/scrypt.pdf>.
- [Pfi01] Birgit Pfitzmann, editor. *EUROCRYPT 2001*, volume 2045 of *LNCS*. Springer, Heidelberg, May 2001.
- [PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008.
- [RG22] Peixin Ren and Xiaozhuo Gu. Practical post-quantum password-authenticated key exchange based-on module-lattice. In Jong Hwan Park and Seung-Hyun Seo, editors, *Information Security and Cryptology - ICISC 2021*, pages 137–156, Cham, 2022. Springer International Publishing.
- [RW04] Renato Renner and Stefan Wolf. The exact price for unconditionally secure asymmetric cryptography. In Cachin and Camenisch [CC04], pages 109–125.
- [RX22] Lawrence Roy and Jiayu Xu. A universally composable pake with zero communication cost (and why it shouldn't be considered uc-secure). Cryptology ePrint Archive, Paper 2022/1607, 2022. <https://eprint.iacr.org/2022/1607>.
- [SAB⁺22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. CRYSTALS-KYBER. Technical report, National Institute

- of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [SGJ23] Bruno Freitas Dos Santos, Yanqi Gu, and Stanislaw Jarecki. Randomized half-ideal cipher on groups with applications to uc (a)pake. Cryptology ePrint Archive, Paper 2023/295, 2023. <https://eprint.iacr.org/2023/295>.
- [srp] SRP protocol design. <http://srp.stanford.edu/design.html>.
- [SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Heidelberg, May 2011.
- [Tha23] Justin Thaler. Proofs, argument, and zero-knowledge, Jan. 24 2023.
- [Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.
- [YGWX20a] Yingshan Yang, Xiaozhuo Gu, Bin Wang, and Taizhong Xu. Efficient password-authenticated key exchange from rlwe based on asymmetric key consensus. In Zhe Liu and Moti Yung, editors, *Information Security and Cryptology*, pages 31–49, Cham, 2020. Springer International Publishing.
- [YGWX20b] Yingshan Yang, Xiaozhuo Gu, Bin Wang, and Taizhong Xu. Efficient password-authenticated key exchange from rlwe based on asymmetric key consensus. In Zhe Liu and Moti Yung, editors, *Information Security and Cryptology*, pages 31–49, Cham, 2020. Springer International Publishing.
- [YKXT20] Yusuke Yoshida, Fuyuki Kitagawa, Keita Xagawa, and Keisuke Tanaka. Non-committing encryption with constant ciphertext expansion from standard assumptions. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 36–65. Springer, Heidelberg, December 2020.
- [You22] Shelanda D. Young. M-23-02 Memo on migrating to post-quantum cryptography, Nov. 18 2022. *Executive Office of the President, Office of Management and Budget*.