Relatively-Sound NIZKs and Password-Based Key-Exchange *

Charanjit Jutla
IBM T. J. Watson Research Center,
Yorktown Heights,
NY 10598, USA

Arnab Roy
Fujitsu Labs of America,
Santa Clara,
CA, USA

Abstract

We define a new notion of relatively-sound non-interactive zero-knowledge (NIZK) proofs, where a private verifier with access to a trapdoor continues to be sound even when the Adversary has access to simulated proofs and common reference strings. It is likely that this weaker notion of relative-soundness suffices in most applications that need simulation-soundness. We show that for certain languages which are diverse groups, and hence allow smooth projective hash functions, one can obtain more efficient single-theorem relatively-sound NIZKs as opposed to simulation-sound NIZKs. We also show that such relatively-sound NIZKs can be used to build rather efficient publicly-verifiable CCA2-encryption schemes.

By employing this new publicly-verifiable encryption scheme along with an associated smooth projective-hash, we show that a recent PAK-model single-round password-based key exchange protocol of Katz and Vaikuntanathan, Proc. TCC 2011, can be made much more efficient. We also show a new single round UC-secure password-based key exchange protocol with only a constant number of group elements as communication cost, whereas the previous single round UC-protocol required $\Omega(k)$ group elements, where k is the security parameter.

*Authors were supported in part by the Department of Homeland Security under grant FA8750-08-2-0091.

Contents

1 Introduction

Authentication based on passwords is a significant security paradigm in today's world. Security in this scenario has been a challenging problem to solve because passwords typically come from low-entropy domains resulting in insufficient randomness for generating cryptographically secure keys. Gong et al. [11] raised the problem of designing protocols resistant to offline password guessing attacks, where other than guessing the low-entropy password by an online attack, the protocol must otherwise provide strong security based on a security parameter. Beginning with the work of Bellovin and Merritt [2], there has been considerable theoretical work in formalizing and obtaining secure protocols in the setting where only passwords are shared by peers (e.g. [1]), referred to as the PAK-security model. From [15] onwards, these protocols employ smooth projective hash functions which have been a standard tool in cryptography ever since Cramer and Shoup defined them to give an efficient chosen ciphertext secure (CCA2) encryption scheme [7].

As illustrated by Gennaro and Lindell [10], who call this the non-malleable commitment paradigm, these protocols require the two peers A and B to non-malleably commit to their password to their peer (say B), e.g. by CCA2 encrypting the password under a public key given as a common reference string (CRS). While, the peer B cannot decrypt this commitment, it might be able to compute a smooth projective-hash on this commitment using a smooth hash key that it generates. The projection of this smooth hash key is sent to peer A, and peer A can compute the same smooth hash using the witness it has for the commitment. The two peers then output a product of two such smooth hashes, one for its own commitment and one for its peer. The problem, however, is that smooth projective-hash for the language, which in this case is the CCA2-ciphertext encrypting a password, is not easy to define, and [10] requires an adaptive smooth hash key, which makes the key-exchange protocol a multi-round protocol.

Recently, Katz and Vaikuntanathan [16] gave a single round protocol for password-based authenticated key exchange, by utilizing a publicly-verifiable CCA2-encryption scheme of Sahai [19]. A publicly-verifiable encryption scheme allows a (non-interactive) public verification of well-formedness of the ciphertext, i.e. it returns TRUE if and only if the decryption oracle will not return an "invalid ciphertext" response when queried with this ciphertext. The public verification allows the smooth hash to be defined on only a part of the ciphertext, which in [16] happens to be two El-Gamal encryptions of the password. Such smooth projective hashes are easy to define and compute.

While the resulting protocol requires only a constant number of group elements, as it employs simulation-sound Groth-Sahai NIZKs [13], under the decisional linear assumption (DLIN [3]) it still requires each party to send 65 group elements (and the run-time is proportionately high).

In this paper we show that the above scheme can be made much more efficient by using a novel concept of *relatively-sound* NIZKs rather than using simulation-sound NIZKs. Simulation-Sound NIZKs were first defined by Sahai [19], where it was used to convert Naor-Yung [18] CCA1-encryption scheme into the aforementioned CCA2-encryption scheme. In simulation-sound NIZKs the NIZK (public) verifier continues to be sound even when the Adversary is given the simulated CRS and proofs. We notice that in most applications what is really required is that a (private) verifier with access to a trapdoor continues to be sound in the simulated world, as long as this private verifier is equivalent to the public verifier in the real-world. The novel relatively-sound NIZKs captures this idea[1].

While it is easy to check that relative-soundness suffices in Sahai's original proof, in this paper we consider a further optimized construction. We prove that an augmented El-Gamal encryption scheme (reminiscent of [8]), along with a labeled single-theorem relatively-sound NIZK leads to a publicly-verifiable CCA2-encryption scheme. In the augmented El-Gamal scheme the public key (under the DDH or SXDH assumptions) consists of g, g^a, g^k, and the encryption of m with randomness x is $g^x, g^{ax}, m \cdot g^{kx}$. The labeled relatively-sound NIZK proves that the first two elements of the ciphertext use the same randomness x, with the third element used as label.

While a single-theorem simulation-sound NIZK could also have been used above, we show that one can obtain single-theorem relatively-sound NIZK far more cheaply than simulation-sound NIZK for this language. We use the fact that the language is a finite diverse group, and hence allows simple 2-universal projective hash functions [7], which allows us to build a private verifier. Under the SXDH assumption [13], converting a NIZK for this language to a relatively-sound NIZK only requires two more group elements, whereas the best-known simulation-sound extension would require nine group elements. Similarly, under the DLIN assumption, our extension requires only three more elements, whereas a simulation-sound extension requires at least 18 more elements [16]. Overall under the DLIN assumption, our publicly-verifiable CCA2 ciphertexts have only 19 group elements versus the 47 group elements in the Sahai scheme [19].

[1] Relatively-sound NIZKs can be considered a hybrid of designated-verifier simulation-sound NIZKs [9] and simulation-sound NIZKs.

We show that using the new encryption scheme in the PAKE protocol of [16], leads to a new protocol which is two to three times more efficient (under both SXDH and DLIN assumptions), with the SXDH-based scheme requiring only 10 group elements to be communicated[2].

UC Security. Canetti et al. [6] proposed a definition of security for password-based key exchange protocols within the Universally Composable (UC) security framework [5], which has the benefit of the universal composition theorem and as such can be deployed as a part of larger security contexts. In addition, their definition of security considers the case of arbitrary and unknown password distributions.

Katz and Vaikuntanathan [16] also gave a single round UC-secure protocol for password-based authenticated key exchange. However, their single round UC protocol is still inefficient as it uses general purpose NIZKs (for NP languages), and further requires proof of knowledge NIZKs. Even if the language for which zero knowledge proofs are required can be made to be given by simple algebraic relations in bilinear groups, the proof of knowledge for exponents of elements as required in their protocol makes it rather expensive.

A second main contribution of this paper is an efficient UC-secure single-round protocol for password based key exchange. The main new ideas required for this efficient protocol are as follows: (a) The shared secret key is obtained in the target group of the bilinear pairings used in the NIZKs which allows for efficient simulator-extraction of group elements corresponding to the smooth-hash trapdoor keys. Such an extraction is required for UC-simulatability. (b) The NIZK proof of knowledge (for extraction) requires the NIZKs to be unbounded simulation-sound. A general construction for unbounded simulation-soundness was given in [4] which is based on a construction due to Groth [12], both of which can be seen to be using relative-soundness implicitly. This leads us to give an optimized version of this general construction. (c) We continue to use the Damgard style [8] encryption scheme, which allows for even more optimization of the unbounded simulation-sound construction for this specific language.

As a result, we get a single-round UC-secure protocol, where under the DLIN-assumption, each party only communicates 63 group elements, which is as efficient as the PAK-model protocol described in [16]. Under the SXDH assumption, our UC-secure protocol only requires 33 group elements.

[2]It should be remarked that other efficient publicly-verifiable CCA2-encryption schemes, such as [17], which allow hash proofs on the (proof-less) part of the ciphertext can also be used in [16].

For sake of exposition, we focus on giving complete proofs only under the SXDH assumption. All of the protocols are also given under the DLIN assumption in the Appendix.

2 NIZK Definitions

In this section we give some definitions related to Non Interactive Zero Knowledge (NIZK) proofs. We will assume familiarity with usual definitions of NIZKs (see e.g. [19, 13]). A proof for a relation R consists of a key generation algorithm K which produces the CRS ψ, a probabilistic polynomial time (PPT) prover P and a PPT verifier V.

Zero-Knowledge. We call (K, P, V) a **NIZK** proof for R if there exists a poly-time simulator (S_1, S_2), such that for all non-uniform PPT adversaries $\mathcal{A}$ we have $\Pr[\psi \leftarrow K(1^m) : \mathcal{A}^{P(\psi,\cdot,\cdot)}(\psi) = 1] \approx \Pr[(\sigma, \tau) \leftarrow S_1(1^m) : \mathcal{A}^{S(\sigma,\tau,\cdot,\cdot)}(\sigma) = 1]$,
where $S(\sigma, \tau, x, w) = S_2(\sigma, \tau, x)$ for $(x, w) \in R$ and both oracles output failure if $(x, w) \notin R$.

One-time Simulation Soundness A NIZK proof is one-time simulation sound NIZK if for all non-uniform PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we have
$\Pr[(\sigma, \tau) \leftarrow S_1(1^m); (x, s) \leftarrow \mathcal{A}_1(\sigma); \pi \leftarrow S_2(\sigma, \tau, x); (x', \pi') \leftarrow \mathcal{A}_2(x, \pi, \sigma, s) :$
$((x', \pi') \neq (x, \pi))$ and $\neg\exists w'$ s.t. $(x', w') \in R$, and $V(\sigma, x', \pi') = 1] \approx 0$.

Unbounded Simulation Sound Extractability (uSS-NIZK). Consider a NIZK proof (K, P, V, S_1, S_2) along with an initialization algorithm SE_1 and a knowledge extractor E_2, such that SE_1 outputs (σ, τ, ξ) with (σ, τ) identical to values output by S_1. Such a proof is said to have the Unbounded Simulation Sound Extractability property if for all non-uniform PPT adversaries $\mathcal{A}$ we have
$\Pr[(\sigma, \tau, \xi) \leftarrow \mathrm{SE}_1(1^k); (x, \pi) \leftarrow \mathcal{A}^{S_2(\sigma,\tau,\cdot)}(\sigma); w \leftarrow E_2(\sigma, \xi, x, \pi) \ :$
$(x, \pi) \notin Q$ and $(x, w) \notin R$ and $V(\sigma, x, \pi) = 1] \approx 0$
where Q is the set of simulation queries and responses (x_i, π_i). For some subset of witnesses the extractor E_2 may extract witnesses in polynomial time, which will be the focus in this paper.

2.1 Relative Soundness

We now define a novel *weaker notion of simulation soundness*, which might suffice for most applications, especially in the case of single theorem (or one-time) simulation. It is possible that this weaker notion may be more efficient to implement, as we demonstrate later for a particularly important

language, where we also show that the weaker notion suffices for the application at hand. In a nutshell, the weaker notion allows for the simulator to have a private verifier of its own, with access to a trapdoor. Simulation-soundness is now defined with respect to simulator's private verifier, and hence the name *relative-soundness*. There is an important further stipulation in the definition that the zero-knowledge property should hold even when the Adversary is given oracle access to private verifier in the simulated world (and public verifier in real world).

Labeled Single-Theorem Relatively-Sound NIZK (l-SRS-NIZK). Consider a sound and complete (labeled) proof (K, P, V) for a relation R along with a PPT private-verifier W and a PPT simulator (S_1, S_2). In a labeled proof, the prover P takes an input label, in addition to the statement to be proven. The verifier takes a statement, a label, and a proof. Such a proof is called a **labeled single-theorem relatively-sound NIZK** for R if for all non-uniform PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4)$ we have

relative-ZK:

$$\Pr[(\psi) \leftarrow K(1^m);\ \ (x, w, \mathtt{lbl}, s) \leftarrow \mathcal{A}_1^{V(\psi,\cdot,\cdot,\cdot)}(\psi);\ \ \pi \leftarrow P(\psi, x, w, \mathtt{lbl}) : \mathcal{A}_2^{V(\psi,\cdot,\cdot,\cdot)}(\pi, s) = 1] \approx$$
$$\Pr[(\sigma, \tau) \leftarrow \mathrm{S}_1(1^m);\ \ (x, w, \mathtt{lbl}, s) \leftarrow \mathcal{A}_1^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\sigma);\ \ \pi \leftarrow S_2(\sigma, \tau, x, \mathtt{lbl}) : \mathcal{A}_2^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\pi, s) = 1],$$

for A_1 restricted to producing (x, w) satisfying R, **and**

relative-simulation-soundness:

$$\Pr[(\sigma, \tau) \leftarrow \mathrm{S}_1(1^m); (x, \mathtt{lbl}, s) \leftarrow \mathcal{A}_3^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\sigma); \pi \leftarrow S_2(\sigma, \tau, x, \mathtt{lbl}); (x', \mathtt{lbl}', \pi') \leftarrow \mathcal{A}_4^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\pi, s) :\ \ ((x', \mathtt{lbl}', \pi') \neq (x, \mathtt{lbl}, \pi)) \text{ and } \neg\exists w' \text{ s.t. } R(x', w') = 1, \text{ and } W(\sigma, \tau, x', \mathtt{lbl}', \pi') = 1] \approx 0.$$

Note that there is no other requirement on W other that those listed above. It is critical that relative-ZK is required only w.r.t. adversaries $(\mathcal{A}_1)$ that produce language members. Otherwise, relative-simulation-soundness would already imply normal simulation-soundness. Although it remains an open problem whether relatively-sound NIZKs are *strictly* weaker than simulation-sound NIZKS, the following shows the relation to non-adaptive simulation soundness, i.e. where the statements for which the proofs need to be simulated are chosen randomly.

2.1.1 Relation to Simulation-Soundness

In this section we consider non-adaptive simulation-soundness, i.e. where the statements for which the proofs need to be simulated are chosen randomly.

Thus, consider the following variant of One-time Simulation Soundness defined in Section 2.

Non-Adaptive Labeled One-time Simulation Soundness A NIZK proof for language $L \subseteq X$ is a non-adaptive one-time simulation sound NIZK if for all non-uniform PPT adversaries $\mathcal{A} = (\mathcal{A}_3, \mathcal{A}_4)$ we have

$$\begin{aligned}\Pr[(\sigma,\tau) \leftarrow \mathrm{S}_1(1^m); x \xleftarrow{\$} X; (\mathtt{lbl}, s) \leftarrow \mathcal{A}_3(\sigma, x); \pi \leftarrow S_2(\sigma, \tau, x, \mathtt{lbl});\\ (x', \mathtt{lbl}', \pi') \leftarrow \mathcal{A}_4(\pi, s) : ((x', \mathtt{lbl}', \pi') \neq (x, \mathtt{lbl}, \pi))\\ \text{and } \neg\exists w'\, R(x', w') = 1, \text{ and } V(\sigma, x', \mathtt{lbl}', \pi') = 1] \approx 0.\end{aligned}$$

Now, assume that the language L is *efficiently witness-samplable*, i.e. there is PPT machine which can efficiently sample from L along with the witness for the language member. Also, a language L, subset of a domain X, is called *hard* if no PPT adversary can distinguish between a (uniformly) random element of L from a random element of X.

For hard and efficiently witness-samplable languages we show that relative-soundness implies non-adaptive simulation-soundness.

Lemma 1 *For a hard and efficiently witness samplable language L, a labeled 1-SRS-NIZK for L also satisfies the non-adaptive labeled one-time simulation soundness property for L.*

Proof: Assume to the contrary that the 1-SRS-NIZK for L does not satisfy the the non-adaptive simulation-soundness property, and let the probability of the event in the definition be a non-negligible Δ. By, the relative-SS property it follows that

$$\begin{aligned}&\Pr[(\sigma,\tau) \leftarrow \mathrm{S}_1(1^m); x \xleftarrow{\$} X; (\mathtt{lbl}, s) \leftarrow \mathcal{A}_3(\sigma, x); \pi \leftarrow S_2(\sigma, \tau, x, \mathtt{lbl});\\ &\quad (x', \mathtt{lbl}', \pi') \leftarrow \mathcal{A}_4(\pi, s) : ((x', \mathtt{lbl}', \pi') \neq (x, \mathtt{lbl}, \pi)) \text{ and } \neg\exists w' \text{ s.t. } R(x', w') = 1,\\ &\quad \text{and } V(\sigma, x', \mathtt{lbl}', \pi') = 1 \text{ and } W(\sigma, \tau, x', \mathtt{lbl}', \pi') = 0] \approx \Delta. \qquad (1)\end{aligned}$$

Now, consider the simulation world where S_1 generates a CRS σ. Also consider an adversary $\mathcal{A}'_1(\sigma)$ which just generates a random $x \in X$, and uses $\mathcal{A}_3(\sigma, x)$ to get label $\mathtt{lbl}$ and state s. Further consider an (oracle) adversary $\mathcal{A}_2(\pi, s)$ which first uses $\mathcal{A}_4(\pi, s)$ to get $x', \mathtt{lbl}', \pi'$, and then applies $V(\sigma, x', \mathtt{lbl}', \pi')$ and equates it with the oracle response on input $x', \mathtt{lbl}', \pi'$. If equality holds $\mathcal{A}_2$ outputs 0, otherwise it outputs 1. It follows from Equation (1) that

$$\begin{aligned}&\Pr[(\sigma,\tau) \leftarrow \mathrm{S}_1(1^m); (x, \mathtt{lbl}, s) \leftarrow \mathcal{A}'_1(\sigma); \pi \leftarrow S_2(\sigma, \tau, x, \mathtt{lbl}) :\\ &\quad A_2^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\pi, s) = 1] > \Delta. \qquad (2)\end{aligned}$$

Now, consider an adversary $\mathcal{A}_1$ which is identical to $\mathcal{A}'_1$ except that it samples x from L instead of X. Then, by hardness of the language L, it follows that (from Equation (2) and the fact that $S_1, S_2, W, V, \mathcal{A}_3, \mathcal{A}_4$ are all PPT machines)

$$\Pr[(\sigma,\tau) \leftarrow \mathrm{S}_1(1^m); (x, \mathtt{lbl}, s) \leftarrow \mathcal{A}_1(\sigma); \pi \leftarrow S_2(\sigma,\tau,x,\mathtt{lbl}) : \mathcal{A}_2^{W(\sigma,\tau,\cdot,\cdot,\cdot)}(\pi, s) = 1] > \Delta.$$

Next, since L is efficiently witness samplable, we let $\mathcal{A}_1$ also generate the witness w of x along with x. Since state s is independent of w, the above probability remains same. Now, by relative-ZK property of 1-SRS-NIZK it follows that

$$\Pr[(\psi) \leftarrow \mathrm{K}(1^m); (x, w, \mathtt{lbl}, s) \leftarrow \mathcal{A}_1(\psi); \pi \leftarrow P(\psi,x,w,\mathtt{lbl}) : \mathcal{A}_2^{V(\psi,\cdot,\cdot,\cdot)}(\pi, s) = 1] > \Delta.$$

Now, $\mathcal{A}_2$ outputs 1 only if $V(\psi,\cdot,\cdot,\cdot)$ (internally) applied to some triple is *not* same as response of oracle on the same triple. But, since the oracle itself is the same $V(\psi,\cdot,\cdot,\cdot)$ the above non-zero probability leads to a contradiction. □

3 Smooth Projective Hash Functions

Fix a cyclic group $G = \langle g, \cdot \rangle$ of prime order q, such that $1/q$ is a negligible function of the security parameter. We define the El-Gamal encryption function as follows. For K, m in G, and x, define

$$\mathsf{enc}^{\mathsf{eg}}_K(m; x) \;=\; \langle g^x, K^x \cdot m \rangle$$

For K and pwd in G, define

$$L_{K,\mathrm{pwd}} = \{c = \langle R, P\rangle \mid \exists x : c = \mathsf{enc}^{\mathsf{eg}}_K(\mathrm{pwd}; x)\} \cap G \times G.$$

A **projective hash function** [7] is a keyed family of functions mapping elements in some message space X to the group G, and is associated with a language. Further, it comes with a **projection function** $\alpha : \mathcal{K} \to S$, where $\mathcal{K}$ is the key space and S is the projected key space. For our hash family, the key space is $\mathbb{Z}_q \times \mathbb{Z}_q$, and the projected key space is G. The message space X is the space of ciphertexts. For $n, \hat{n}$ in $\mathbb{Z}_q$, c in G^2, and K, pwd in G, define the hash family $\mathcal{H}^{K,\mathrm{pwd}}$ associated with $L_{K,\mathrm{pwd}}$ by

$$\mathcal{H}^{\mathrm{pwd}}_{n,\hat{n}}(c = \langle R, P\rangle) = (P/\mathrm{pwd})^{\hat{n}} \cdot R^n, \qquad \alpha^{K,\mathrm{pwd}}(n, \hat{n}) = g^n \cdot (K)^{\hat{n}}.$$

It is straightforward to see that, if $c = \mathsf{enc}^{\mathsf{eg}}_K(\mathrm{pwd}; x)$ for some x, then $\mathcal{H}^{\mathrm{pwd}}_{n,\hat{n}}(c) = \alpha^{K,\mathrm{pwd}}(n, \hat{n})^x$.

For any K and pwd in G, $\mathcal{H}^{K,\mathrm{pwd}}$ is said to be **smooth** [7] w.r.t. $L = L_{K,\mathrm{pwd}}$, if for any c' in G^2, but *not* in L, the statistical distance between the distribution of the pair $(\mathcal{H}^{K,\mathrm{pwd}}_{n,\hat{n}}(c'),\ \alpha^{K,\mathrm{pwd}}(n.\hat{n}))$ and the pair (g^{d_1}, g^{d_2}) is negligible, where $n, \hat{n}, d_1, d_2$ are chosen randomly and independently from $\mathbb{Z}_q$. It is a simple exercise to see that $\mathcal{H}^{K,\mathrm{pwd}}$ is smooth with respect to $L_{K,\mathrm{pwd}}$.

We also define a projective hash function family associated with any language L to be **2-universal** [7] if for all $s \in S$, $x, x' \in X$, and $\pi, \pi' \in G$ with $x \notin L \cup \{x'\}$, it holds that $\Pr_k[H_k(x) = \pi \mid H_k(x') = \pi' \wedge \alpha(k) = s] \leq 1/q$.

4 Bilinear Assumptions

Throughout the paper, we use (bilinear) groups G_1, G_2, G_T each of prime order q, which allow an efficiently computable $\mathbb{Z}_q$-bilinear pairing map $e : G_1 \times G_2 \to G_T$.

SXDH: [13] The symmetric external decisional Diffie-Hellman (SXDH) assumption states that the decisional Diffie-Hellman (DDH) problem is hard in both groups G_1 and G_2.

DLIN: [3] In groups such that G_1 is same as G_2, the decisional linear (DLIN) assumption states that given $(\alpha\mathcal{P}, \beta\mathcal{P}, r\alpha\mathcal{P}, s\beta\mathcal{P}, t\mathcal{P})$ for random $\alpha, \beta, r, s \in \mathbb{Z}_q$, and arbitrary generator $\mathcal{P}$ of G_1, it is hard to distinguish between $t = r + s$ and a random t.

5 A Publicly-Verifiable CCA2-Encryption Scheme

In this section we describe a CCA2-Encryption scheme that has the property that a potential ciphertext can be publicly verified to be a valid ciphertext of some message. Note that Sahai [19] had previously given a publicly-verifiable CCA2-encryption scheme employing the Naor-Yung CCA1-scheme [18], but our scheme is simpler and more efficient.

One might be tempted to take the Cramer-Shoup encryption scheme, and extend the ciphertext by including a NIZK proof that the 2-universal smooth projective-hash [7] was correctly computed. However, since the

NIZK scheme by itself may be malleable, this may render the scheme insecure in the CCA2-model. There are two potential fixes to this: (a) make the NIZK single theorem simulation-sound, or (b) include the NIZK commitments to the witness in the projective-hash. While it is not that difficult to see that (a) may lead to a correct publicly-verifiable CCA2-scheme (just as in [19]), the second idea (b) may seem far-fetched.

We now show that it suffices to make the NIZK proof a labeled single-theorem *relatively-sound* NIZK, and further one just needs to prove in this NIZK that the Diffie-Hellman tuple in the ciphertext is well-formed, i.e. it is of the form g^x, A^x. We later show that there exists a very efficient way to extend a single-theorem Groth-Sahai NIZK of this statement to be a relatively-sound proof, such that the resulting publicly-verifiable CCA2-scheme is just the idea (b) mentioned above.

To formally define publicly-verifiable CCA2-encryption schemes, one just extends the standard IND-CCA2 definition of encryption with a public verification function V which takes the public key and a potential ciphertext as arguments, and it returns true iff the decryption function when supplied with the same ciphertext does not return "invalid ciphertext".

For given g, A, let the relation $\mathcal{R} = \{((\rho, \hat{\rho}), x) \mid \rho = g^x, \hat{\rho} = A^x\}$. We now define a *labeled* publicly-verifiable public-key encryption scheme DHENC as follows:

Key Generation: Generate $g, A \xleftarrow{\$} G_1$, and $k \xleftarrow{\$} \mathbb{Z}_q$. Let $K = g^k$. Let ψ be the CRS for an l-SRS-NIZK. The public key is (g, A, K, ψ) and the private key is k.

Encrypt: Given plaintext $m \in G_1$, and label `lbl`. Choose $x \xleftarrow{\$} \mathbb{Z}_q$. Let the triple $\langle \rho, \hat{\rho}, \gamma \rangle$ be $\langle g^x, A^x, mK^x \rangle$. Let π be an l-SRS-NIZK proof of $((\rho, \hat{\rho}), x) \in \mathcal{R}$ with label $\gamma, \texttt{lbl}$. The ciphertext is $(\rho, \hat{\rho}, \gamma, \pi)$.

Decrypt: Given ciphertext $c = (\rho, \hat{\rho}, \gamma, \pi)$ and label `lbl`. Verify if π is an l-SRS-NIZK proof for $(\rho, \hat{\rho})$ and label $\gamma, \texttt{lbl}$. If verification fails output $\perp$. Otherwise output $m = \frac{\gamma}{\rho^k}$.

Verify: Given ciphertext $c = (\rho, \hat{\rho}, \gamma, \pi)$ and label `lbl`. Verify if π is an l-SRS-NIZK proof for $(\rho, \hat{\rho})$ and label $\gamma, \texttt{lbl}$. If verification fails output false else output true.

Theorem 2 *The scheme* DHENC *is publicly-verifiable (labeled) IND-CCA2 secure.*

The full proof of this theorem can be found in the Appendix, but the main idea is that the decryption can be done as either γ/ρ^k, or as $\gamma/(\rho^{k'}\hat{\rho}^{k''})$, where the Simulator chooses the public key K as $g^{k'}A^{k''}$. The encryption oracle hides the message by employing DDH as follows: (1) The NIZK CRS in the original experiment is the binding-CRS, and the decryption oracle in the original experiment does a public verification of proofs in each adversarially supplied ciphertext. (2) The NIZK CRS is switched to be the hiding CRS, the proof switched to a simulator generated proof, and decryption oracle now uses private-verification. This is an indistinguishable change by the relative-ZK property of l-SRS-NIZK. Note, x is no more used in the simulated proof. (3) The decryption is done as $\gamma/(\rho^{k'}\hat{\rho}^{k''})$, which is equivalent because of relative-simulation soundness property of l-SRS-NIZK. (4) DDH is employed, as only g^a instead of a is being used in simulation. This leads to A^x being replaced by an independent X'. (5) The decryption is done as γ/ρ^k, which is again equivalent by relative-soundness. (6) the message in the encryption can be switched by pairwise independence in k, and this step is information-theoretic. More precisely, $g^{xk'}(X')^{k''}$ is random and independent of g^x, X', K, A, as well as Adversary's coins with high probability. (7) Next we do all the above steps (2)-(5) in reverse.

6 l-SRS-NIZK for the DDH Language

Let G_1 and G_2 be two groups with a bilinear pairing $e : G_1 \times G_2 \to G_T$ and $|G_1| = |G_2| = |G_T| = q$, a prime number. Also assume that DDH is hard for both G_1 and G_2. Recall that this is the `SXDH` assumption. Let $L_{g,A}$ be the language: $\{(\rho, \hat{\rho}) \in {G_1}^2 \mid \exists x.\ \rho = g^x \wedge\ \hat{\rho} = A^x\}$, with g, A in G_1.

Note that this language is actually a cyclic group with generator $\langle g, A\rangle$, and forms a diverse group system [7]. In [7], Cramer and Shoup show how to obtain 2-universal projective hash functions for such languages, and we use these hash functions for private-verification.

We construct an l-SRS-NIZK proof system for $L_{g,A}$, as follows:

CRS Generation: Generate $\mathcal{P} \xleftarrow{\$} G_2$ and $u, v, d_1, d_2, e_1, e_2 \xleftarrow{\$} \mathbb{Z}_p$. Compute $(P, Q, R,\ S,\ \mathbf{d},\ \mathbf{e}) = (\mathcal{P},\ \mathcal{P}^u,\ \mathcal{P}^v,\ \mathcal{P}^{uv+1},\ g^{d_1}A^{d_2},\ g^{e_1}A^{e_2})$. The CRS is $\psi = (P, Q, R, S,\ \mathbf{d},\ \mathbf{e})$. The first four elements are as in the Groth-Sahai NIZK for SXDH (*binding* CRS), and the last two are the projection keys for a 2-universal projective-hash for the DDH language (just as [7]), to be used in the relatively-sound system.

The simulation CRS σ is $(P, Q, R,\ S,\ \mathbf{d},\ \mathbf{e}) = (\mathcal{P},\ \mathcal{P}^u,\ \mathcal{P}^v,\ \mathcal{P}^{uv},\ g^{d_1}A^{d_2},$

$g^{e_1}A^{e_2}$). This is the *hiding* CRS of GS-NIZK for SXDH along with $\mathbf{d}$ and $\mathbf{e}$ as above. The trapdoor is $\tau = (u, d_1, d_2, e_1, e_2)$.

Prover: Given witness x, candidate (g^x, A^x), and label $\mathtt{lbl}$, construct proof as follows. Generate $s \xleftarrow{\$} \mathbb{Z}_q$. Compute $t \leftarrow H(g^x, A^x, Q^x P^s, S^x R^s, \mathtt{lbl})$, where H is a collision resistant hash function. Then compute: $(\beta, c_1, c_2, \theta, \phi, \chi) \leftarrow \left((\mathbf{de}^t)^x, Q^x P^s, S^x R^s, g^s, A^s, (\mathbf{de}^t)^s\right)$. Output proof $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$. The first element is a 2-universal projective-hash computed on the candidate with witness x. The last five elements can be interpreted as generated by the Groth-Sahai NIWI proof (which also happens to be a NIZK proof) for the language $\{\rho, \hat{\rho}, h \mid \exists x \ : \rho = g^x, \hat{\rho} = A^x, h = (\mathbf{de}^t)^x\}$, where t is a hash of ρ, $\hat{\rho}$, $\mathtt{lbl}$, *and the commitment* to x in the NIWI system, i.e. $Q^x P^s, S^x R^s$.

Simulator: Given a candidate $(\rho, \hat{\rho})$, generate the proof as follows. Generate $s \xleftarrow{\$} \mathbb{Z}_q$ and compute $t \leftarrow H(\rho, \hat{\rho}, P^s, R^s, \mathtt{lbl})$. Then compute

$$\pi = (\beta, c_1, c_2, \theta, \phi, \chi) = \left(\rho^{d_1}\hat{\rho}^{d_2}(\rho^{e_1}\hat{\rho}^{e_2})^t, P^s, R^s, \rho^{-u}g^s, \hat{\rho}^{-u}A^s, \beta^{-u}(\mathbf{de}^t)^s\right)$$

Public Verify: Given $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$ as a candidate proof of $(\rho, \hat{\rho})$ with label $\mathtt{lbl}$, compute $t \leftarrow H(\rho, \hat{\rho}, c_1, c_2, \mathtt{lbl})$. Then check the following equations:

$$\begin{pmatrix} e(g, c_1) \stackrel{?}{=} e(\rho, Q) \cdot e(\theta, P), & e(g, c_2) \stackrel{?}{=} e(\rho, S) \cdot e(\theta, R) \\ e(A, c_1) \stackrel{?}{=} e(\hat{\rho}, Q) \cdot e(\phi, P), & e(A, c_2) \stackrel{?}{=} e(\hat{\rho}, S) \cdot e(\phi, R) \\ e(\mathbf{de}^t, c_1) \stackrel{?}{=} e(\beta, Q) \cdot e(\chi, P), & e(\mathbf{de}^t, c_2) \stackrel{?}{=} e(\beta, S) \cdot e(\chi, R) \end{pmatrix}$$

Private Verify: Given $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$ as a candidate proof of $(\rho, \hat{\rho})$ with label $\mathtt{lbl}$, compute $t \leftarrow H(\rho, \hat{\rho}, c_1, c_2, \mathtt{lbl})$. Then first do public verification and if that succeeds then check the following equation: $\beta \stackrel{?}{=} \rho^{d_1}\hat{\rho}^{d_2}(\rho^{e_1}\hat{\rho}^{e_2})^t$.

Note that this private verifier is well-defined in the real world as well. In addition, its trapdoor (d_1, d_2, e_1, e_2) is identically generated in both the real and the simulated worlds.

Theorem 3 *The above system is an l-SRS-NIZK proof system for $L_{g,A}$.*

A detailed proof of this theorem can be found in the appendix, but the following proof sketch is instructive.

Proof Sketch: We focus on Relative-ZK and Relative-SS properties. For the former, we need to show that the simulation CRS, and a proof for $(\rho, \hat{\rho})$ with label `lbl` is computationally indistinguishable from the real CRS and a real proof, even when the Adversary has oracle access to respective veriifers. This is accomplished by a sequence of games, where the first game is same as te real world game. In the second game, the CRS and the proof remain the same but the verifier in the oracle is changed to be the private verifier. We need to show that public verification implies private verification, but this follows from soundness of the Groth-Sahai NIZK, as well as the fact that on a *valid* DDH tuple the projection hash is same whether it is computed using the witness and the projection key or using the private hash keys. In the final game we switch to the simulation CRS and simulated proof, and indistinguishability follows from ZK property of Groth-Sahai NIZKs and the fact that the private verification trapdoor is independent of the Groth-Sahai NIZK CRS (hiding or binding).

The relative-simulation-soundness property is proven using the 2-universal property of the projective smooth hash (just as in [7]), but additionaly using the fact that in Groth-Sahai NIZKs, once the commitments to the witnesses are fixed, there is a unique proof satisfying the linear equations of the type used in the above NIZK proof. This holds for both the SXDH and the DLIN assumptions. □

The l-SRS-NIZK proof for DDH language above consists of six group elements. The l-SRS-NIZK proof for the DLIN language (and under the DLIN assumption), given in the appendix, consists of 15 group elements.

7 Secure Protocol in the PAK Model

In this section we present a password-based key exchange protocol secure in the PAK model of security due to Bellare, Pointcheval and Rogaway [1]. We instantiate the single-round scheme due to Katz and Vaikuntanathan [16], which is described in Figure 1, with the more efficient publicly-verifiable CCA-secure encryption scheme DHENC of Section 5, which enables a more efficient hash proof as well. The common reference string (CRS) is just the public key of this scheme. More details can be found in the Appendix.

The projective-hash family used in this scheme is $\mathcal{H}^{\mathrm{pw}}$ along with the projection function $\alpha^{K,\mathrm{pw}}$ defined in Section 3, where K is from the public-key (i.e. CRS). Note that the input *label* to the hash function is ignored in $\mathcal{H}^{\mathrm{pw}}$. Also, α does not depend on pw.

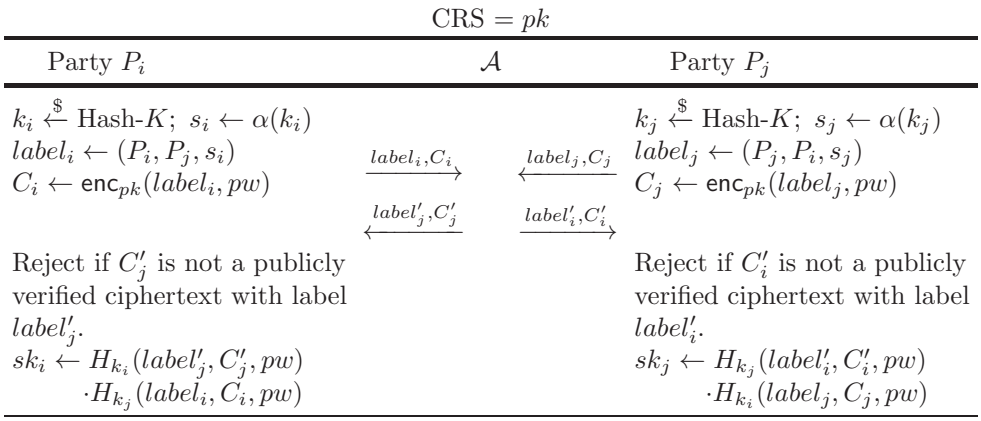

Figure 1: Single-round PAK-Model Secure Password-based Authenticated KE.

Theorem 4 *Assume the existence of SXDH-hard groups G_1 and G_2. Then the protocol in Figure 1 is secure in the PAK model.*

The proof of this theorem is same as the proof in [16], as we have modularized the various constructs required in that proof. The main idea is that once the CCA2-encryption scheme is publicly verifiable, then the smooth hash needs to be just over the language $L_{K,\mathrm{pw}}$, which are CPA encryptions of password. However, we provide an alternate stand-alone proof in Appendix C.2.

8 Secure Protocol in the UC Model

8.1 Universally Composable Security

The Universally Composable (UC) framework [5] is a formal system for proving security of computational systems such as cryptographic protocols. The framework describes two probabilistic games: The *real world* that captures the protocol flows and the capabilities of an attacker, and the *ideal world* that captures what we think of as a secure system. The notion of security asserts that these two worlds are essentially equivalent.

8.2 UC Functionality for Password-Based Key Exchange

The essential elements of the Universal Composability framework can be found in [5]. We adopt the definition for password-based key exchange from

Functionality $\mathcal{F}_{\mathbf{pwKE}}$

The functionality $\mathcal{F}_{\text{pwKE}}$ is parameterized by a security parameter k. It interacts with an adversary S and a set of parties via the following queries:

Upon receiving a query (NewSession, $sid, P_i, P_j, pw, role$) **from party** P_i**:**
Send (NewSession, $sid, P_i, P_j, role$) to S. In addition, if this is the first NewSession query, or if this is the second NewSession query and there is a record (P_j, P_i, pw'), then record (P_i, P_j, pw) and mark this record fresh.

Upon receiving a query (TestPwd, sid, P_i, pw') **from the adversary** S**:**
If there is a record of the form (P_i, P_j, pw) which is fresh, then do: If $pw = pw'$, mark the record compromised and reply to S with "correct guess". If $pw \neq pw'$, mark the record interrupted and reply with "wrong guess".

Upon receiving a query (NewKey, sid, P_i, sk) **from** S**, where** $|sk| = k$**:**
If there is a record of the form (P_i, P_j, pw), and this is the first NewKey query for P_i, then:

- If this record is compromised, or either P_i or P_j is corrupted, then output (sid, sk) to player P_i.
- If this record is fresh, and there is a record (P_j, P_i, pw') with $pw' = pw$, and a key sk' was sent to P_j, and (P_j, P_i, pw) was fresh at the time, then output (sid, sk') to P_i.
- In any other case, pick a new random key sk' of length k and send (sid, sk') to P_i.

Either way, mark the record (P_i, P_j, pw) as completed.

Figure 2: The password-based key-exchange functionality $\mathcal{F}_{\text{pwKE}}$

Canetti et al [6]. The following description is a summary from [6]. The formal description is given in Figure 2.

Like the key exchange functionality, if both participating parties are not corrupted, then they receive the same uniformly distributed session key and the adversary learns nothing of the key except that it was generated. However, if one of the parties is corrupted, then the adversary determines the session key. This power to the adversary is also given in case it succeeds in guessing the parties' shared password. Participants also detect when the adversary makes an unsuccessful attempt. If the adversary makes a wrong password guess in a given session, then the session is marked interrupted and the parties are provided random and independent session keys. If the adversary makes a successful guess, then the session is marked compromised. If a session is marked fresh, this means that it is neither interrupted nor compromised. Such sessions between uncorrupted parties conclude with both parties receiving the same, uniformly distributed session key.

8.3 A Single Round UC Password-Based Key Exchange Protocol

The single-round UC protocol under the SXDH assumption uses labeled unbounded simulation sound G_2-extractable NIZKs (uSS-NIZK). Consider parties P_i and P_j involved in the protocol with SSID ssid. The CRS is three group elements $g, A(= g^a), K(= g^k)$ chosen randomly from G_1, another element $\mathcal{P}$ chosen randomly from G_2, and a uSS-NIZK CRS ψ. Since g, $\mathcal{P}$ are also part of the uSS-NIZK CRS, having chosen the NIZK CRS, g, $\mathcal{P}$ are already determined. The protocol is symmetric and asynchronous with each party computing a message to be sent, then receiving a corresponding message and computing a key. Therefore, we just describe it from the perspective of one party; the other is symmetric.

Party P_i generates $x \xleftarrow{\$} \mathbb{Z}_q$ and computes $c_1 = \langle g^x, A^x, K^x \cdot pw \rangle$. It also generates hash key $(n_1, \hat{n_1}) \xleftarrow{\$} (\mathbb{Z}_q)^2$ and computes the projection key $\eta_1 = \alpha^{K,\text{pwd}}(n_1, \hat{n_1}) = g^n \cdot K^{\hat{n}}$. Finally it computes a NIZK proof of consistency in the following way:

$$\pi_1 = \text{uSS-NIZK}_\psi(g^x, A^x, \eta_1; x, \mathcal{P}^{n_1}, \mathcal{P}^{\hat{n}_1}) \text{ with label } \langle P_i, P_j, \mathsf{ssid} \rangle$$

Note that π here denotes the commitments to the witnesses as well as the further proof as in the Groth-Sahai system. The NP language L for the NIZK is

$$L = \{\rho, \hat{\rho}, \eta \mid \exists x, N, \hat{N} \ : \ \rho = g^x, \hat{\rho} = A^x, \ e(\eta, \mathcal{P}) = e(g, N)e(K, \hat{N})\}$$

Now, the message sent by P_i is $\langle c_1, \eta_1, \pi_1 \rangle$. Let the message received by P_i in this session, supposedly from P_j, be $\langle c_2', \eta_2', \pi_2' \rangle$. Let c_2' be parsed as $(\rho_2', \hat{\rho_2'}, \gamma_2')$. If any of $\rho_2', \hat{\rho}_2', \gamma_2', \eta_2'$ is not in $G_1 \backslash \{1\}$, or uSS-NIZK-Verify$(\pi_2'; \rho_2', \hat{\rho}_2', \eta_2')$ with label $\langle P_j, P_i, \mathsf{ssid} \rangle$ turns out to be false, then it sets its session key sk_1 randomly from the target group of e, G_T. Otherwise it is computed as follows:

$$h_2' = (\frac{\gamma_2'}{\text{pwd}})^{\hat{n}_1}(\rho_2')^{n_1} \qquad h_1 = (\eta_2')^{x_1} \qquad h_3 = h_2' \cdot h_1 \qquad \text{sk}_1 = e(h_3, \mathcal{P}).$$

Theorem 5 *Assume the existence of a SXDH-hard group, a labeled unbounded simulation-sound G_2-extractable NIZK proof system. Then the protocol in Figure 3 securely realizes the $\widehat{\mathcal{F}}_{\text{PWKE}}$ functionality in the $\mathcal{F}_{\mathsf{crs}}$ hybrid model, in the presence of static corruption adversaries.*

$\text{CRS} = g, \mathcal{P}, A, K, \psi : \quad g, A, K \xleftarrow{\$} G_1 \quad \mathcal{P} \xleftarrow{\$} G_2 \quad \psi = \text{uSS-NIZK CRS}$

Party P_i		Adv $\mathcal{A}$
Input $(\texttt{NewSession}, sid, \mathsf{ssid}, P_i, P_j, \text{pwd}, initiator/responder)$		
Choose $x_1, n_1, \hat{n}_1 \xleftarrow{\$} \mathbb{Z}_q$.		
Set $\rho_1 = g^{x_1}$, $\hat{\rho}_1 = (A)^{x_1}$, $\gamma_1 = \text{pwd} \cdot K^{x_1}$, $\eta_1 = g^{n_1}(K)^{\hat{n}_1}$,	$\xrightarrow{c_1, \eta_1, \pi_1}$	$\mathcal{A}$
Let $c_1 = \langle \rho_1, \hat{\rho}_1, \gamma_1 \rangle$, and		
$\pi_1 = \text{uSS-NIZK}_\psi(\rho_1, \hat{\rho}_1, \eta_1; x_1, \mathcal{P}^{n_1}, \mathcal{P}^{\hat{n}_1})$ with label $\langle P_i, P_j, \mathsf{ssid} \rangle$.		
	$\xleftarrow{c_2', \eta_2', \pi_2'}$	$\mathcal{A}$
Let $c_2' = \langle \rho_2', \hat{\rho}_2', \gamma_2' \rangle$.		
If any of $\rho_2', \hat{\rho}_2', \gamma_2', \eta_2'$ is not in $G_1 \backslash \{1\}$, or		
not uSS-NIZK-Verify$(\pi_2; \rho_2', \hat{\rho}_2', \eta_2')$ with label $\langle P_j, P_i, \mathsf{ssid} \rangle$		
set $\text{sk}_1 \xleftarrow{\$} G_T$, else		
compute $h_2' = (\frac{\gamma_2'}{\text{pwd}})^{\hat{n}_1}(\rho_2')^{n_1}$, $h_1 = (\eta_2')^{x_1}$, $\text{sk}_1 = e(h_2' \cdot h_1, \mathcal{P})$.		
Output $(sid, \mathsf{ssid}, \text{sk}_1)$.		

Figure 3: Single round UC-secure Password-based KE under SXDH Assumption.

In the next section we demonstrate a simulator which uses $\widehat{\mathcal{F}}_{\text{PWKE}}$ to simulate the protocol to an adversary, thus proving Theorem 6.

For sake of exposition, we describe the protocol based on the SXDH (symmetric external Diffie-Hellman) assumption. In the appendix, we give the protocol under the Decisional Linear assumption (DLIN). The SXDH based protocol is given in Fig 3, and a detailed description follows.

The protocol uses labeled unbounded simulation sound G_2-extractable NIZKs (uSS-NIZK). Consider parties P_i and P_j involved in the protocol with SSID ssid. The CRS is three group elements $g, A(= g^a), K(= g^k)$ chosen randomly from G_1, another element $\mathcal{P}$ chosen randomly from G_2, and a uSS-NIZK CRS ψ. Since g, $\mathcal{P}$ are also part of the uSS-NIZK CRS, having chosen the NIZK CRS, g, $\mathcal{P}$ are already determined. The protocol is symmetric and asynchronous with each party computing a message to be sent, then receiving a corresponding message and computing a key. Therefore, we just describe it from the perspective of one party; the other is symmetric.

Party P_i generates $x \xleftarrow{\$} Z_q^*$ and computes $c_1 = \langle g^x, A^x, K^x \cdot pw \rangle$. It also generates hash key $(n_1, \hat{n_1}) \xleftarrow{\$} (Z_q^*)^4$ and computes the projection key $\eta_1 = \alpha^{K,\text{pwd}}(n_1, \hat{n_1}) = g^n \cdot K^{\hat{n}}$. Finally it computes a NIZK proof of

consistency in the following way:

$$\pi_1 = \text{uSS-NIZK}_\psi(g^x, A^x, \eta_1; x, \mathcal{P}^{n_1}, \mathcal{P}^{\hat{n}_1}) \text{ with label } \langle P_i, P_j, \mathsf{ssid}\rangle$$

Note that π here denotes the commitments to the witnesses as well as the further proof as in the Groth-Sahai system. The NP language L for the NIZK is

$$L = \{\rho, \hat{\rho}, \eta \mid \exists x, N, \hat{N} \ : \ \rho = g^x, \hat{\rho} = A^x, \ e(\eta, \mathcal{P}) = e(g, N)e(K, \hat{N})\}$$

Now, the message sent by P_i is $\langle c_1, \eta_1, \pi_1\rangle$. Let the message received by P_i in this session, supposedly from P_j, be $\langle c_2', \eta_2', \pi_2'\rangle$. Let c_2' be parsed as $(\rho_2', \hat{\rho_2'}, \gamma_2')$. If any of $\rho_2', \hat{\rho}_2', \gamma_2', \eta_2'$ is not in $G_1\backslash\{1\}$, or uSS-NIZK-Verify$(\pi_2'; \rho_2', \hat{\rho}_2', \eta_2')$ with label $\langle P_j, P_i, \mathsf{ssid}\rangle$ turns out to be false, then it sets its session key sk_1 randomly from the target group of e, G_T. Otherwise it is computed as follows:

$$h_2' = (\frac{\gamma_2'}{\text{pwd}})^{\hat{n}_1}(\rho_2')^{n_1} \qquad h_1 = (\eta_2')^{x_1} \qquad h_3 = h_2' \cdot h_1 \qquad \text{sk}_1 = e(h_3, \mathcal{P}).$$

Theorem 6 *Assume the existence of a SXDH-hard group, a labeled unbounded simulation-sound G_2-extractable NIZK proof system. Then the protocol in Figure 3 securely realizes the $\widehat{\mathcal{F}}_{\text{PWKE}}$ functionality in the $\mathcal{F}_{\mathsf{crs}}$ hybrid model, in the presence of static corruption adversaries.*

In the next section we demonstrate a simulator which uses $\widehat{\mathcal{F}}_{\text{PWKE}}$ to simulate the protocol to an adversary, thus proving Theorem 6.

A more optimized version of such a general labeled unbounded simulation sound G_2-extractable NIZK [7] is given in the Appendix in Section D. In fact, for the language above for which such a NIZK is required, we give a further optimization in Appendix **??**. Based on this optimized construction, the uSS-NIZK requires 29 group elements. A similar construction under the DLIN assumption, and for the DLIN based UC-secure PWKE-construction (see Appendix E.4) requires 54 group elements.

8.4 The Simulator for the UC Protocol

The trapdoor keys a, k for the CRS are chosen differently by the simulator. Instead of choosing a, k randomly from $\mathbb{Z}_q$, the simulator chooses a, k', k'' from $\mathbb{Z}_q$ and sets $k = k' + a \cdot k''$. It outputs $A = g^a$ and $K = g^k = g^{k'}(g^a)^{k''}$ as before. Note that this does not change the distribution of A and K, as

$\mathbb{Z}_q$ is a field. (We will continue to write k for $k' + ak''$, except when the simulation in some experiments needs to be done with g^a, instead of a).

Simulator S also invokes the initialization phase SE_1 of the labeled uSS-NIZK (with security parameter m) to obtain (σ, τ, ξ). S then gives A, K, and σ to the real world adversary $\mathcal{A}$ as the *common reference string*. Thereafter, the simulator S interacts with the environment $\mathcal{Z}$, the functionality $\widehat{\mathcal{F}}_{\text{PWKE}}$, and uses $\mathcal{A}$ as a subroutine. The messages between $\mathcal{Z}$ and $\mathcal{A}$ are just forwarded by S.

The main difference in the simulation of the real world parties is that S uses a dummy message μ instead of the real password which it does not have access to. Further, it generates all proofs using the NIZK simulator S_2 instead of real prover.

8.4.1 New Session: Sending a message to $\mathcal{A}$.

On message (NewSession, *sid*, ssid, i, j, role) from $\widehat{\mathcal{F}}_{\text{PWKE}}$, S starts simulating a new session of the protocol Π for party P_i, peer P_j, session identifier ssid, and CRS $= (A, K, \psi)$. We will denote this session by (P_i, ssid). To simulate this session, S chooses x_1 at random, and sets c_1 $(= \langle \rho_1, \hat{\rho}_1, \gamma_1 \rangle)$ to $\langle g^{x_1}, A^{x_1}, \mu \cdot K^{x_1} \rangle$. It also chooses hash keys $n_1, \hat{n_1}$ at random, and computes the smooth-hash projected key η_1 as in the real protocol as well. S obtains a fake NIZK proof π_1 using the simulator S_2 of the NIZK, and the CRS σ, and simulation trapdoor τ. It then hands c_1, η_1, π_1 to $\mathcal{A}$ on behalf of this session.

More succinctly, the simulator behavior is as follows:

$$x_1, n_1, \hat{n}_1, \xleftarrow{\$} \mathbb{Z}_q^* \tag{3}$$

$$c_1 \ (= \langle \rho_1, \hat{\rho}_1, \gamma_1 \rangle) \ = \langle g^{x_1}, A^{x_1}, \mu \cdot K^{x_1} \rangle \tag{4}$$

$$\eta_1 = \alpha^{K,\mu}(n_1, \hat{n}_1) = g^{n_1 + k\hat{n}_1} \tag{5}$$

$$\pi_1 = \text{uSS-NIZK-SIM}_{\psi}(\rho_1, \hat{\rho}_1, \eta_1) \text{ with label } \langle P_i, P_j, \mathsf{ssid} \rangle. \tag{6}$$

8.4.2 On Receiving a Message from $\mathcal{A}$.

On receiving a message c_2', η_2', π_2' from $\mathcal{A}$ intended for this session (P_i, ssid), the simulator S makes the real world protocol checks including verifying the NIZK proof using the NIZK-verifier. If any of the checks fail, it issues a TestPwd call to $\widehat{\mathcal{F}}_{\text{PWKE}}$ with the dummy password μ, followed by a NewKey call with a random session key, which leads to the functionality issuing a random and independent session key to the party P_i (regardless of whether the session was interrupted or compromised).

Otherwise, it computes pwd′ by decrypting c_2', i.e. setting it to $\gamma_2'/(\rho_2')^k$. If the message received from $\mathcal{A}$ is same as message sent by S on behalf of peer P_j in session ssid, then S just issues a NewKey call for P_i. Otherwise, S calls $\widehat{\mathcal{F}}_{\text{PWKE}}$ with (TestPwd, ssid, P_i, pwd′). Regardless of the reply from $\mathcal{F}$, it then issues a NewKey call for P_i with key computed as follows (*this is different from the real-world protocol.*). This has the effect that if the pwd′ was same as the actual pwd in $\widehat{\mathcal{F}}_{\text{PWKE}}$ then the session key is determined by the Simulator, otherwise the session key is set to a random and independent value. Here is the complete simulator code (stated as it's overall experiment with $\mathcal{Z}$, including $\mathcal{F}$'s communication with $\mathcal{Z}$):

1. Let $c_2' = \langle \rho_2', \hat{\rho}_2', \gamma_2' \rangle$.
2. If any of $\rho_2', \hat{\rho}_2', \gamma_2', \eta_2'$ is not in $G_1 \backslash \{1\}$, or *not* uSS-NIZK-Verify$(\pi_2'; \rho_2', \hat{\rho}_2', \eta_2')$ with label $\langle P_j, P_i, \mathsf{ssid} \rangle$, output $\text{sk}_1 \xleftarrow{\$} G_T$, else compute as follows.
3. If msg rcvd == msg sent in same session (same SSID) by peer, set $\text{sk}_1 \xleftarrow{\$} G_T$, unless the peer also received a legitimate message and its key has already been set, in which case that same key is used to set sk_1.
4. Else, compute $N_2', \hat{N}_2'$ from the proof π_2', using the extraction trapdoor ξ.
5. Compute $\text{pwd}' = \gamma_2'/(\rho_2')^k$. If (pwd′ ≠ pwd) then $\text{sk}_1 \xleftarrow{\$} G_T$, else
6. $h_2' = (\frac{\gamma_2'}{\text{pwd}'})^{\hat{n}_1}(\rho_2')^{n_1}$, $h_1 = (\eta_2')^{x_1}$; set $\text{sk}_1 = e(h_2', \mathcal{P}) \cdot e(h_1, \mathcal{P}) \cdot e(\mu/\text{pwd}, \hat{N}_2')$.

Note that the main difference is the additional factor $e(\mu/\text{pwd}, \hat{N}_2')$.

8.5 Proof of Indistinguishability for the UC Protocol

We now describe a series of experiments between the Simulator and the environment, starting with Expt_0 which is the same as the experiment described as the Simulator in Section 8.4 above, and ending with an experiment which is identical to the real world execution of the protocol in Fig 3. We will show that the environment has negligible advantage in distinguishing between these experiments, leading to a proof of realization of $\mathcal{F}_{\text{PWKE}}$ by the protocol Π.

For each instance, we will use subscript 2 along with a prime, to refer to variables after the reception of the message from $\mathcal{A}$, and use subscript 1 to refer to variables computed before sending the message to $\mathcal{A}$. We will call a message legitimate if it was not altered by the adversary, and delivered in the correct session, and to the correct party.

Expt_1**:** The experiment Expt_1 is same as Expt_0 except for the following modified step 3 in the reception code: *If msg rcvd == msg sent in same session by peer, set* sk_1 to

$$e(\mathcal{H}^{\text{pwd}}_{n_1,\hat{n}_1}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mu; x_2)) \cdot \mathcal{H}^{\text{pwd}}_{n_2,\hat{n}_2}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mu; x_1)), \mathcal{P}).$$

Because the hash proof system is for languages with messages encrypting real password, the smooth-hash-proof yields random values from the adversary's point of view. Note that we only employ the hash proof system corresponding to n_1 and $\hat{n}_1$, and note that the second factor corresponding to n_2 and $\hat{n}_2$ is independent of the first. In step 6, n_1 and $\hat{n}_1$ are being used, but the code never gets there if the msg received is same as message sent by legitimate peer.

Expt_2: Next, we replace all occurrences of $e(h_1, \mathcal{P})$ $(= e((\eta_2')^{x_1}, \mathcal{P}))$ in the computation of sk_1 in Step 6 of the reception code by $e(g, N_2')^{x_1} \cdot e(K, (\hat{N}_2')^{x_1})$, which is the same as $e(g^{x_1}, N') \cdot e(K^{x_1}, \hat{N}')$. This leads to an indistinguishable change as the simulator had verified the NIZK proofs, and the NIZK proofs have unbounded simulation extractability property, and thus $e(\eta_2', \mathcal{P}) = e(g, N_2')e(K, \hat{N}_2')$.

Expt_3: The next change in simulation is to replace μ by the real password in the outgoing message element γ. However, since the simulator is employing k to compute pwd', one cannot directly employ DDH to replace μ by pwd in outgoing γ. However, since we are using an augmented El-Gamal encryption scheme, i.e. also including $\hat{\rho}$ in the outgoing message along with a proof of its relation to ρ, we can use the pairwise independence in k to accomplish our goal, just as in CCA2 scheme DHENC described in Section 5.

At this point, not only is the outgoing γ_1 being computed as $K^{x_1} \cdot \text{pwd}$, i.e. $c_1 = \mathsf{enc}^{\mathsf{eg}}_K(\text{pwd}; x_1)$, but also in the reception phase of the same (ssid, P_i), the term $e(\mu/\text{pwd}, \hat{N}_2')$ has been replaced by 1. Recall that in Expt_2, $e(h_1, \mathcal{P})$ was replaced by $e(g^{x_1}, N') \cdot e(K^{x_1}, \hat{N}')$, and now $e(K^{x_1}, \hat{N}')$ has been replaced by $e(\text{pwd}/\mu \cdot K^{x_1}, \hat{N}')$, which is then equivalent to replacing $e(\mu/\text{pwd}, \hat{N}_2')$ by 1 in Step 6. Further, if the message received was legitimate, then sk_1 is now set to

$$e(\mathcal{H}^{\text{pwd}}_{n_1,\hat{n}_1}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mu; x_2)) \cdot \mathcal{H}^{\text{pwd}}_{n_2,\hat{n}_2}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\text{pwd}; x_1), \mathcal{P}).$$

Similarly, if the peer received a legitimate message, its computation of sk_1 has a similar change, i.e. its first factor has μ replaced by pwd. Thus, at the end of these sequence of hybrid experiments, if the message received was legitimate, then sk_1 is now set to $e(\mathcal{H}^{\mathrm{pwd}}_{n_1,\hat{n}_1}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mathrm{pwd}; x_2)) \cdot \mathcal{H}^{\mathrm{pwd}}_{n_2,\hat{n}_2}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mathrm{pwd}; x_1)), \mathcal{P})$.

Expt_4: In this experiment we drop the condition *if (pwd′ ≠ pwd) then set sk_1 to random* in Step 5, and always output as follows

$$h'_2 = (\frac{\gamma'_2}{\mathrm{pwd}})^{\hat{n}_1/\mathsf{ssid}}(\rho'_2)^{n_1},\ h_1 = (\eta'_2)^{x_1}; \text{set } \mathrm{sk}_1 = e(h'_2, \mathcal{P}) \cdot e(g^{x_1}, N'_2) \cdot e(K^{x_1}, \hat{N}'_2).$$

This is accomplished by a series of hybrid experiments, one for each (ssid, P_i), we employ the hash proof smoothness, as $\mathrm{pwd}' \neq \mathrm{pwd}$ implies the tuple c'_2 is not in the language, and hence h'_2 is anyway random and independent.

Expt_5: In this experiment we set sk_1 in the last step as $e(h'_2, \mathcal{P}) \cdot e(\eta'^{x_1}_2, \mathcal{P})$. This change is indistinguishable as the simulator is checking the validity of the NIZK proofs, and by simulation-soundness extractability.

Expt_6: In this experiment we can drop the extraction of N'_2 and $\hat{N}'_2$, as they are no longer needed, and further we drop step 3. Note that currently that step is computing sk_1 as $e(\mathcal{H}^{\mathrm{pwd}}_{n_1,\hat{n}_1}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mathrm{pwd}; x_2)) \cdot \mathcal{H}^{\mathrm{pwd}}_{n_2,\hat{n}_2}(\mathsf{enc}^{\mathsf{eg}}_{g^k}(\mathrm{pwd}; x_1)), \mathcal{P})$, but since $\eta'_2 = \eta_2$, and $c'_2 = c_2$ for this session, then the above expression is same as $e(h'_2, \mathcal{P}) \cdot e(\eta'^{x_1}_2, \mathcal{P})$. We replace all simulator generated proofs by proofs generated by real prover, and switch from the CRS generated by SE_1 to the real world CRS. Experiment Expt_6 is indistinguishable from the real-world experiment by completeness of the hash proof system, i.e. when the labeled tuple c, ssid is in the language, then the hash can be computed from the projection keys and the witness x_1 of c. This completes the proof of Theorem 6. □

References

[1] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 139–155. Springer, May 2000. 1, 7, C.1

[2] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *1992 IEEE Symposium on Security and Privacy*, pages 72–84. IEEE Comp. Soc. Press, May 1992. 1

[3] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Aug. 2004. 1, 4

[4] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 351–368. Springer, Apr. 2009. 1, D, 1, 2, D

[5] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Comp. Soc. Press, Oct. 2001. 1, 8.1, 8.2

[6] R. Canetti, S. Halevi, J. Katz, Y. Lindell, and P. D. MacKenzie. Universally composable password-based key exchange. In *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 404–421. Springer, May 2005. 1, 8.2

[7] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Apr. / May 2002. 1, 3, 5, 6, 6, 8.3, B, E, E.1

[8] I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Aug. 1992. 1

[9] E. Elkind and A. Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attacks. Cryptology ePrint Archive: Report 2002/042. 1

[10] R. Gennaro and Y. Lindell. A framework for password-based authenticated key exchange. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 524–543. Springer, May 2003. `http://eprint.iacr.org/2003/032.ps.gz`. 1

[11] L. Gong, T. M. A. Lomas, R. M. Needham, and J. H. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE JSAC*, 11(5):648–656, June 1993. 1

[12] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Dec. 2006. 1

[13] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Apr. 2008. 1, 2, 4, B, 1, D

[14] C. Jutla and A. Roy. Relatively-sound NIZKs and password-based key-exchange. Cryptology ePrint Archive, Report 2011/507, 2011. `http://eprint.iacr.org/`. D

[15] J. Katz, R. Ostrovsky, and M. Yung. Efficient password-authenticated key exchange using human-memorable passwords. In *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 475–494. Springer, May 2001. 1, C.1

[16] J. Katz and V. Vaikuntanathan. Round-optimal password-based authenticated key exchange. In *TCC 2011*, volume 6597 of *LNCS*, pages 293–310. Springer, Mar. 2011. 1, 2, 7, 7, 9, E.3

[17] E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC 2006*, volume 3876 of *LNCS*, pages 581–600. Springer, Mar. 2006. 2

[18] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*. ACM Press, May 1990. 1, 5

[19] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Comp. Soc. Press, Oct. 1999. 1, 2, 5

A Appendix: Publicly-Verifiable CCA2 Encryption

Theorem 2 The scheme DHENC is publicly-verifiable IND-CCA2 secure.

The definition of IND-CCA2 security is now standard and can be found in [**?**]. In the labeled version [**?**], the Adversary calls the encryption oracle with two messages m_0, and m_1 and a label `lbl`. The encryption oracle returns c, an encryption of m_b with label `lbl`, with b chosen randomly. The Adversary can make arbitrary decryption calls, except that after receiving the encryption c, it cannot call the decryption oracle with c and the same label `lbl`. It is however allowed to call the decryption oracle with the same

ciphertext c, but a different label $\texttt{lbl}'$. The notion of security is the advantage of the adversary in guessing b.

Proof: The public-verification part is straightforward, and we focus on the IND-CCA2 security part. We demonstrate a sequence of experiments showing indistinguishability of an encryption of an arbitrary m with an encryption of a fixed dummy message μ also in the message space. All experiments are identical to the previous experiment except for the noted modifications.

$\mathbf{Expt}_1$: This is the same as the real protocol, where if a ciphertext and label pair, i.e. $(\rho, \hat{\rho}, \gamma, \pi, \texttt{lbl})$ is same as encryption query output and encryption query input label, then decryption oracle does not decrypt. Otherwise, decryption of a message $(\rho, \hat{\rho}, \gamma, \pi)$ with label $\texttt{lbl}$, after verifying the l-SRS-NIZK proof π with label $(\gamma, \texttt{lbl})$ (using public-verifier V), is $m = \frac{\gamma}{\rho^k}$.

$\mathbf{Expt}_2$: In this experiment, the simulator chooses $k', k'' \xleftarrow{\$} \mathbb{Z}_q$ and sets $K = g^{k'} A^{k''}$ in the public key. It decrypts a message $(\rho, \hat{\rho}, \gamma, \pi)$ with label $\texttt{lbl}$, after public verifying π with label $(\gamma, \texttt{lbl})$, as $m = \frac{\gamma}{\rho^{k'} \hat{\rho}^{k''}}$.

The (regular) soundness of the l-SRS-NIZK implies that $\hat{\rho} = \rho^a$ whp and hence $\rho^k = \rho^{k'} \hat{\rho}^{k''}$. Also the distribution of k remains the same. $\mathbf{Expt}_2$ is therefore indistinguishable from $\mathbf{Expt}_1$.

$\mathbf{Expt}_3$: In this experiment, the simulator switches to a l-SRS-NIZK CRS generated by S_1 and simulated proof of membership of (g^x, A^x) generated by S_2. Simulator retains the trapdoor τ. Further, the public verification $V(\psi, ...)$ in each decryption request is replaced by private verification $W(\sigma, \tau, ...)$.

Note that, given challenge plaintext m, it now returns the ciphertext $c = \Big(g^x, A^x, g^{k'x} A^{k''x} \cdot m, \text{Simulated } \pi\Big)$. Because of the switching, x is not needed as a witness in the l-SRS-NIZK proof.

$\mathbf{Expt}_3$ is indistinguishable from $\mathbf{Expt}_2$ by the relative-ZK property of the l-SRS-NIZK, noting that g^x, A^x are in the language of the NIZK.

$\mathbf{Expt}_4$: In this experiment, the simulator replaces A^x by a fresh random value X' from the group $\mathcal{G}$. Further, instead of choosing x from $\mathbb{Z}_q^*$ and using g^x in the challenge ciphertext, we directly choose X randomly from $\mathcal{G}$. Thus the Simulator code in this experiment is the following:

$k', k'' \xleftarrow{\$} \mathbb{Z}_q^*$; $A \xleftarrow{\$} \mathcal{G}$; $K = g^{k'} A^{k''}$; $\boxed{X, X' \xleftarrow{\$} \mathcal{G}}$;
$c = \Big(X, \boxed{X', X^{k'}(X')^{k''} \cdot m}, \text{Simulated } \pi\Big)$.
Public Key output is $(g, A, K, S_1$ generated 1-SRS-NIZK CRS $\sigma)$. Challenge Ciphertext output is c. Decryption queries are served by private-verifying the proof and then outputing $\frac{\gamma}{\rho^{k'} \hat{\rho}^{k''}}$.

$\mathbf{Expt}_4$ is computationally indistinguishable from $\mathbf{Expt}_3$ due to the DDH assumption, as in the previous experiment X' was $A^{\log X}$ (further, choosing X randomly from $\mathcal{G}$ is same as choosing x from $\mathbb{Z}_q^*$ and computing g^x, as $\mathcal{G}$ is a prime-order cyclic group).

$\mathbf{Expt}_5$: In this experiment, the simulator chooses a at random from $\mathbb{Z}_q^*$ and sets A to g^a. Now, the simulator decrypts as follows: on a message $(\rho, \hat{\rho}, \gamma, \pi)$ and label $\texttt{lbl}$, after private-verifying the l-SRS-NIZK π with label $(\gamma, \texttt{lbl})$, output $m = \frac{\gamma}{\rho^{k'+ak''}}$. Thus the Simulator code in this experiment is the following:

$k', k'' \xleftarrow{\$} \mathbb{Z}_q^*$; $\boxed{a \xleftarrow{\$} \mathbb{Z}_q^*;\ A = g^a}$; $K = g^{k'} A^{k''}$; $X, X' \xleftarrow{\$} \mathcal{G}$;
$c = \Big(X, X', X^{k'}(X')^{k''} \cdot m, \text{Simulated } \pi\Big)$.
Public Key output is $(g, A, K, S_1$ generated 1-SRS-NIZK CRS $\sigma)$. Challenge Ciphertext output is c. Decryption queries are served by private-verifying the proof and then outputing $\boxed{\frac{\gamma}{\rho^{k'+ak''}}}$.

Note that the Experiment continues to ignore decryption queries if $(\rho, \hat{\rho}, \gamma, \pi, \texttt{lbl})$ is same as encryption query output and encryption query input label pair. Thus, the l-SRS-NIZK relative-simulation-soundness property implies that $\hat{\rho} = \rho^a$ whp and hence $\rho^{k'+ak''} = \rho^{k'} \hat{\rho}^{k''}$. Technically, this experiment should be broken up into several sub-experiments, where in each sub-experiment an additional decryption request is served by outputting $m = \gamma/\rho^{k'+ak''}$ (instead of $\gamma/\rho^{k'} \hat{\rho}^{k''}$). Since the adversary (in particular $\mathcal{A}_3$ and $\mathcal{A}_4$) in definition of l-SRS-NIZK has access to private-verification oracle, one can employ relative-simulation-soundness for each sub-experiment. Further distribution of A is same as in the previous experiment as $\mathcal{G}$ is a prime-order cyclic group. Thus, $\mathbf{Expt}_5$ is indistinguishable from $\mathbf{Expt}_4$.

$\mathbf{Expt}_6$: In this experiment, in response to encryption oracle request on m, the challenge ciphertext becomes $c = \Big(X, X', X^{k'}(X')^{k''} \cdot \mu, \text{Simulated } \pi\Big)$, i.e. m is replaced by μ.

Decryption of $(\rho, \hat{\rho}, \gamma, \pi)$, after verifying the l-SRS-NIZK, is same as before, i.e. $\frac{\gamma}{\rho^{k'+ak''}}$.

Thus the Simulator code in this experiment is the following:

$k', k'' \xleftarrow{\$} \mathbb{Z}_q^*$; $a \xleftarrow{\$} \mathbb{Z}_q^*$; $A = g^a$; $K = g^{k'} A^{k''}$; $X, X' \xleftarrow{\$} \mathcal{G}$;
$c = \Big(X, X', \boxed{X^{k'}(X')^{k''} \cdot \mu}, \text{Simulated } \pi\Big)$.
Public Key output is $(g, A, K, S_1$ generated 1-SRS-NIZK CRS $\sigma)$. Challenge Ciphertext output is c. Decryption queries are served by private-verifying the proof and then outputing $\frac{\gamma}{\rho^{k'+ak''}}$.

Lemma 7 ***Expt***$_6$ *is statistically indistinguishable from* ***Expt***$_5$.

Proof: First note that in both $\mathbf{Expt}_5$ and $\mathbf{Expt}_6$, the quantities k', k'', a, X, X' are chosen randomly and independently of each other as well as all Adversary inputs (the first three from $\mathbb{Z}_q^*$ and the latter two from $\mathcal{G}$). Further note that in both experiments, the decryption queries can be served by $\log K = k' + ak''$. Let *coins* be the random coins used by the Adversary. Thus, if we show that for all $m, \mu \in \mathcal{G}$, the distribution of $(X, X', A, K, X^{k'}(X')^{k''} \cdot m, coins)$ is statistically indistinguishable from the distribution of $(X, X', A, K, X^{k'}(X')^{k''} \cdot \mu, coins)$ under a, k', k'' chosen randomly from $\mathbb{Z}_q^*$, and X, X' chosen randomly from $\mathcal{G}$, and *coins* chosen randomly, then we are done (simulated proof π does not need any witnesses; technically we should also include private verification keys σ, τ also in the joint distribution, but they are same in the two experiments and the rest of the elements above are chosen independently of them).

Let g be any fixed generator of $\mathcal{G}$. In the following we will use x as log of X w.r.t. g, and similarly x' as log of X'. Now with k', k'' and a chosen randomly from $\mathbb{Z}_q^*$, and X, X' chosen randomly from $\mathcal{G}$, for all $m, \mu \in \mathcal{G}$, and for all $\alpha, \kappa, \chi, \chi', \delta \in \mathcal{G}$, and c in domain of *coins*, we have

$$\begin{aligned}
&\Pr[(X, X', K, A, coins) = \langle \chi, \chi', \kappa, \alpha, c\rangle \ \wedge\ X^{k'}(X')^{k''} \cdot m = \delta] \\
&= \Pr[(g^x, g^{x'}, g^{k'+ak''}, g^a, coins) = \langle \chi, \chi', \kappa, \alpha, c\rangle \ \wedge\ g^{xk'+x'k''} \cdot m = \delta] \\
&= \Pr[g^{k'+ak''} = \kappa \ \wedge\ g^{xk'+x'k''} \cdot m = \delta \mid (g^x, g^{x'}, g^a, coins) = \langle \chi, \chi', \alpha, c\rangle] \ * \\
&\quad \Pr[(g^x, g^{x'}, g^a, coins) = \langle \chi, \chi', \alpha, c\rangle]
\end{aligned}$$

We first focus on the case that $\log \chi' \neq (\log \chi * \log \alpha)$. Then, in this case for a fixed value of x, x', a and *coins* (which are $\log \chi$, $\log \chi'$, $\log \alpha$, c resp.), the probability that $k' + \log \alpha \cdot k''$ is $\log \kappa$ and $\log \chi \cdot k' + \log \chi' \cdot k''$ is $\log(\delta/m)$ is exactly $1/|\mathcal{G}|^2$, as the two terms are linearly independent.

Since this probability is same for any δ and any m, the probability is also the same for any δ and any μ.

Since the statistical difference between $\mathbf{Expt}_5$ and $\mathbf{Expt}_6$ is at most the sum over all $\alpha, \kappa, \chi, \chi', \delta \in \mathcal{G}$ (and all c) of the absolute value of the difference in the above probabilities, it is seen to be at most the probability that $g^{x'} = g^{x*a}$ (i.e. summing over all χ, χ', α such that $\log \chi' = (\log \chi * \log \alpha)$, as in other cases the distributions are the same), which is just $1/q$. Hence, the statistical difference between $\mathbf{Expt}_5$ and $\mathbf{Expt}_6$ is at most $1/q$, which is negligible.

□

$\mathbf{Expt}_7$: In this experiment, the simulator decrypts as follows: On a message $(\rho, \hat{\rho}, \gamma, \pi)$ and label $\texttt{lbl}$, after private-verifying the l-SRS-NIZK with label $\gamma, \texttt{lbl}$, output $\frac{\gamma}{\rho^{\tilde{k}'} \hat{\rho}^{\tilde{k}''}}$.

Thus the Simulator code in this experiment is the following:

> $k', k'' \xleftarrow{\$} \mathbb{Z}_q^*$; $a \xleftarrow{\$} \mathbb{Z}_q^*$; $A = g^a$; $K = g^{k'} A^{k''}$; $X, X' \xleftarrow{\$} \mathcal{G}$;
> $c = \Big(X, X', X^{k'} (X')^{k''} \cdot \mu, \text{Simulated } \pi\Big)$.
> Public Key output is $(g, A, K, S_1$ generated 1-SRS-NIZK CRS $\sigma)$. Challenge Ciphertext output is c. Decryption queries are served by private-verifying the proof and then outputing $\boxed{\frac{\gamma}{\rho^{k'} \hat{\rho}^{k''}}}$.

Note that the Experiment continues to ignore decryption queries if $(\rho, \hat{\rho}, \gamma, \pi, \texttt{lbl})$ is same as encryption query output and encryption query input label pair. Say, the encryption query output on input label $\texttt{lbl}^*$ is $(\rho^*, \hat{\rho}^*, \gamma^*, \pi^*)$. Let the i-th decryption query be $(\rho_i, \hat{\rho}_i, \gamma_i, \pi_i, \texttt{lbl}_i)$. If $(\rho_i, \hat{\rho}_i, \pi_i, \langle \gamma_i, \texttt{lbl}_1 \rangle)$ is same as $(\rho^*, \hat{\rho}^*, \pi^*, \langle \gamma^*, \texttt{lbl}^* \rangle)$ then the decryption query ignores this decryption request. Since $\langle \gamma, \texttt{lbl} \rangle$ was the label used in the NIZK proof generation, it follows from the l-SRS-NIZK relative-simulation-soundness property that $\hat{\rho} = \rho^a$ whp and hence $\rho^k = \rho^{\tilde{k}'} \hat{\rho}^{\tilde{k}''}$. Also the distribution of k remains the same. $\mathbf{Expt}_7$ is therefore indistinguishable from $\mathbf{Expt}_6$.

$\mathbf{Expt}_8$: In this experiment, instead of choosing X randomly from $\mathcal{G}$, the simulator chooses x randomly from $\mathbb{Z}_q^*$ and sets $X = g^x$, and further simulator replaces X' back by A^x.

Thus the Simulator code in this experiment is the following:

$k', k'' \xleftarrow{\$} \mathbb{Z}_q^*$; $a \xleftarrow{\$} \mathbb{Z}_q^*$; $A = g^a$; $K = g^{k'} A^{k''}$; $\boxed{x \xleftarrow{\$} \mathbb{Z}_q^*;\ X = g^x; X' = A^x}$;
$c = \Big(X, X', X^{k'}(X')^{k''} \cdot \mu, \text{Simulated } \pi\Big)$.
Public Key output is $(g, A, K, S_1$ generated 1-SRS-NIZK CRS $\sigma)$. Challenge Ciphertext output is c. Decryption queries are served by private-verifying the proof and then outputing $\frac{\gamma}{\rho^{k'}\hat{\rho}^{k''}}$.

$\mathbf{Expt}_8$ is indistinguishable from $\mathbf{Expt}_7$ due to the DDH assumption. Let the DDH challenge be $g^x, A\ (= g^a), X'\backslash A^x$. In experiment $\mathbf{Expt}_7$, the Simulator uses the first component of the DDH challenge tuple, i.e. g^x, for X, second component for A, and the third component of the tuple for X'. In experiment $\mathbf{Expt}_8$ also, the Simulator uses the first component, i.e. g^x, for X, second component for A, and the third component of the tuple for X'. Thus, if an efficient adversary can distinguish $\mathbf{Expt}_7$ from $\mathbf{Expt}_8$, the Simulator can break the DDH challenge.

$\mathbf{Expt}_9$: In this experiment, the simulator switches back to the real CRS, and proofs generated by real prover with witness x. Therefore the response to the encryption challenge m becomes
$c = \Big(g^x, A^x, g^{\tilde{k}'x} A^{\tilde{k}''x} \cdot \mu, \pi\Big)$.
Further, the decryption queries now use public verifier instead of private-verifier. $\mathbf{Expt}_9$ is indistinguishable form $\mathbf{Expt}_8$ by the relative-ZK property of l-SRS-NIZK, noting that g^x, A^x is in the language of the NIZK.

$\mathbf{Expt}_{10}$: In this experiment, the simulator just uses k instead of $\tilde{k}', \tilde{k}''$. Therefore the response to the encryption challenge m becomes $c = (g^x, A^x, K^x \cdot \mu, \pi)$. Decryption of a message $(\rho, \hat{\rho}, \gamma, \pi)$, after (public) verifying the l-SRS-NIZK, is $m = \frac{\gamma}{\rho^k}$.

The (regular) soundness of the l-SRS-NIZK implies that $\hat{\rho} = \rho^a$ whp and hence $\rho^k = \rho^{\tilde{k}'}\hat{\rho}^{\tilde{k}''}$. Also the distribution of k remains the same. $\mathbf{Expt}_{10}$ is therefore indistinguishable from $\mathbf{Expt}_9$.

This proves that DHENC is CCA2 secure because the protocol is indistinguishable from Experiment 10, in which everything is same except that in the response to an encryption request a constant message is encrypted, instead of the request message.

□

B Appendix: Proof of l-SRS-NIZK

Theorem 3: The system in Section 6 is an l-SRS-NIZK proof system for $L_{g,A}$.

Proof:

We will refer to d_1, d_2, e_1, e_2 as the private-verifier trapdoor ξ.

Completeness and Soundness. Since the first four tests in the Public Verification above are same as the tests in the Groth-Sahai NIWI for the language $L_{g,A}$, soundness follows from soundness of the NIWI system [13]. Completeness follows by simple verification.

Relative Zero-Knowledge. We construct a simulator as follows. Generate $\mathcal{P} \xleftarrow{\$} G_2$; and $u, v, d_1, d_2, e_1, e_2 \xleftarrow{\$} \mathbb{Z}_p$. Set CRS $\sigma = (P, Q, R, S, \mathbf{d}, \mathbf{e}) = (\mathcal{P}, \mathcal{P}^u, \mathcal{P}^v, \boxed{\mathcal{P}^{uv}, g^{d_1}A^{d_2}, g^{e_1}A^{e_2}})$. The simulation trapdoor τ is $\boxed{u, d_1, d_2, e_1, e_2}$, i.e. u and the private-verifier trapdoor ξ.

Given a candidate $(\rho, \hat{\rho})$ and label `lbl`, generate the proof as follows. Generate $s \xleftarrow{\$} \mathbb{Z}_p$ and compute $t \leftarrow H(\rho, \hat{\rho}, P^s, R^s, \texttt{lbl})$. Then compute

$$\pi = (\beta, c_1, c_2, \theta, \phi, \chi) = \left(\rho^{d_1}\hat{\rho}^{d_2}(\rho^{e_1}\hat{\rho}^{e_2})^t, P^s, R^s, \rho^{-u}g^s, \hat{\rho}^{-u}A^s, \beta^{-u}(\mathbf{d}\mathbf{e}^t)^s\right)$$

We now show that this CRS and a proof for $(\rho, \hat{\rho}) = (g^x, A^x) \in L_{g,A}$ with label `lbl` is computationally indistinguishable from a real CRS and a real proof for the same candidate and label, given the DDH assumption for G_2, even when the Adversary has oracle access to respective verifiers. To this end, we enumerate a sequence of games each indistinguishable from the previous starting from a real CRS and a real proof and ending at the simulated CRS and a simulated proof:

Game 1: This is the real CRS and real proof:

$$\text{CRS } \psi = (P, P^u, P^v, P^{uv+1}, \mathbf{d}, \mathbf{e})$$

Proof for $(x : \rho, \hat{\rho}) = (x : g^x, A^x)$ with label `lbl` can be seen to be computed as follows:

Generate $s \leftarrow \mathbb{Z}_p$
Compute $c_1 \leftarrow P^{ux+s}, c_2 \leftarrow (P^{uv+1})^x(P^v)^s$
Compute $t \leftarrow H(\rho, \hat{\rho}, c_1, c_2, \texttt{lbl})$
Compute $\beta \leftarrow (\mathbf{d}\mathbf{e}^t)^x, \theta \leftarrow g^s, \phi \leftarrow A^s, \chi \leftarrow (\mathbf{d}\mathbf{e}^t)^s$
Proof $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$

Game 2: In this game, the CRS and the proof remains the same as real CRS and real proof, but the verifier in the oracle is changed to be the private verifier. Thus, this requires that the key generation algorithm generate a trapdoor, which is set to (u, d_1, d_2, e_1, e_2), i.e. u along with ξ. Indistinguishablity of Game 1 and 2 will follow if we show that the private verifier and public verifier are equivalent.

Since the private verifier also tests the public verifier, the success of private verification implies the success of public verification. We now argue that the converse also holds. That is the success of public verification implies the success of private verification.

Observe that β is a 2-universal projective hash computed on the candidate with witness x. From the projection property of the projective hash function, on a valid DDH tuple the projective hash is same whether it is computed using the witness or the projection keys (ξ)– this is straightforward to check.

The elements $c_1, c_2, \theta, \phi, \chi$ can be interpreted as generated by the Groth-Sahai NIWI proof (which also happens to be a NIZK proof) for the language $\{(\rho, \hat{\rho}, \beta) \mid \exists x \ : \rho = g^x, \hat{\rho} = A^x, \beta = (\mathbf{de}^t)^x\}$, where t is a hash of ρ, $\hat{\rho}$, `lbl`, *and the commitment* to x in the NIWI system, i.e. c_1, c_2. Hence the soundness of this NIZK implies that β has the correct form, that is, private verification would succeed. This is indeed the case as the NIZK CRS is the binding CRS of the GS-NIZK, which assures soundness of public verifier.

Game 3: In this game the CRS is modified as follows:

$$\text{CRS } \sigma = (P, P^u, P^v, \boxed{P^{uv}}, \mathbf{d}, \mathbf{e})$$

By DDH for G_2, this is computationally indistinguishable from Game 2 (note the verification trapdoor ξ is independent of the DDH tuple).

Proof for $(x : \rho, \hat{\rho}) = (x : g^x, A^x)$ is computed identically as Game 2, except it uses the CRS σ:

Generate $s \leftarrow \mathbb{Z}_p$
Compute $c_1 \leftarrow P^{ux+s}, c_2 \leftarrow (P^{uv})^x (P^v)^s$
Compute $t \leftarrow H(\rho, \hat{\rho}, c_1, c_2, \texttt{lbl})$
Compute $\beta \leftarrow (\mathbf{de}^t)^x, \theta \leftarrow g^s, \phi \leftarrow A^s, \chi \leftarrow (\mathbf{de}^t)^s$
Proof $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$

Game 4: In this game the NIZK CRS remains σ. The simulator is given the trapdoor u, d_1, d_2, e_1, e_2.

The proof is computed as follows:

Generate $s' \leftarrow \mathbb{Z}_p$

Compute $c_1 \leftarrow P^{s'}, c_2 \leftarrow (P^v)^{s'}$

Compute $t \leftarrow H(\rho, \hat{\rho}, c_1, c_2, \mathtt{lbl})$

Compute $\beta \leftarrow \rho^{d_1}\hat{\rho}^{d_2}(\rho_1^{e_1}\hat{\rho}^{e_2})^t, \theta \leftarrow \rho^{-u}g^{s'}, \phi \leftarrow \hat{\rho}^{-u}A^{s'}, \chi \leftarrow \beta^{-u}(\mathbf{de}^t)^{s'}$

NIZK Proof $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$

The proof for a *language member* $(x : \rho, \hat{\rho}) = (x : g^x, A^x)$ with label $\mathtt{lbl}$ is identical to Game 3, except that it uses the variable $s' = ux + s$. Observe that s' is distributed uniformly given uniform distribution of s. Also $g^s = \rho^{-u}g^{s'}$ and $A^s = \hat{\rho}^{-u}A^{s'}$.

Indistinguishability from Game 3 is information theoretic. Since Game 4 is just the simulator game, we are through.

Labeled Single-theorem Relative-Simulation-Soundness. Suppose the adversary is provided a simulated proof $\pi = (\beta, c_1, c_2, \theta, \phi, \chi)$ for a language candidate $(\rho, \hat{\rho})$ and label $\mathtt{lbl}$ chosen by the adversary. We have to show that any candidate, label and proof $(\rho', \hat{\rho'}, \mathtt{lbl}', \pi') \neq (\rho, \hat{\rho}, \mathtt{lbl}, \pi)$ provided by the adversary such that $(\rho', \hat{\rho'}) \notin L_{g,A}$, will fail the private verification test with high probability. Let $\pi' = (\beta', c_1', c_2', \theta', \phi', \chi')$ and $t' = H(\rho', \hat{\rho}', c_1', c_2', \mathtt{lbl}')$. We analyze two cases:

Case $t' \neq t$: If the private verification succeeds, then we have the following linear equations, where $A = g^a, \rho = g^{x_1}, \hat{\rho} = g^{x_2}, \rho' = g^{x_1'}, \hat{\rho}' = g^{x_2'}$:

$$\begin{pmatrix} 1 & a & 0 & 0 \\ 0 & 0 & 1 & a \\ x_1 & x_2 & tx_1 & tx_2 \\ x_1' & x_2' & t'x_1' & t'x_2' \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \\ e_1 \\ e_2 \end{pmatrix} = \begin{pmatrix} \log \mathbf{d} \\ \log \mathbf{e} \\ \log \beta \\ \log \beta' \end{pmatrix}$$

Observe that if $t' \neq t$ and $x_2' \neq ax_1'$, then the left-hand-side four by four matrix is full-ranked. Note, d_1, d_2, e_1, e_2 are chosen randomly and independently. However, the adversary (i.e. $\mathcal{A}_3$ and $\mathcal{A}_4$) has oracle access to the private-verifier W. Just as in [7], we show that a polynomial number p of oracle accesses to W does not substantially increase adversary's chances of passing private verification on a non-language tuple. We prove the claim by induction on the number of

oracle accesses of $\mathcal{A}_3$ and $\mathcal{A}_4$. In the base case, when there are no oracle accesses, $\log \beta'$ is uniformly distributed even conditioned on $\log \mathbf{d}, \log \mathbf{e}, \log \beta$. Since the latter three constitute the only information available to the adversary about d_1, d_2, e_1, e_2, the adversary succeeds in providing the correct β' only with negligible probability.

Next, consider the case where $\mathcal{A}_3$ and $\mathcal{A}_4$ make p oracle calls. By induction, we can assume that these calls are either with language elements or simulator generated proof or the oracle W responds negatively. In the first two cases the adversary gets linear equations in d_1, d_2, e_1, e_2 which are linear combinations of the first three equations above. In the last case, adversary gets an inequality. Already due to the first three equations above, d_1, d_2, e_1, e_2 is restricted to be a random point on a line in $(\mathbb{Z}_q)^4$. The inequality, rules out one point on this line. Thus, a maximum of p such inequalities rule out p points from a total of q points on the line. Since $q = 2^k$ where, k is the security parameter, the probability of the fourth equation holding above is $1/(2^k - p)$.

Case $t' = t$: In this case, by the collision resistance property of the hash, whp $(\rho', \hat{\rho}', c_1', c_2', \mathtt{lbl}') = (\rho, \hat{\rho}, c_1, c_2, \mathtt{lbl})$. If private verification succeeds (which includes public verification), then

$$\begin{aligned}
&\beta' = \rho^{d_1} \hat{\rho}^{d_2} \left(\rho^{e_1} \hat{\rho}^{e_2}\right)^t = \beta \\
&e(\theta', P) = e(g, c_1) \cdot e(\rho, Q)^{-1} = e(\theta, P) \\
&e(\phi', P) = e(A, c_1) \cdot e(\hat{\rho}, Q)^{-1} = e(\phi, P) \\
&e(\chi', P) = e(\mathbf{d}\mathbf{e}^t, c_1) \cdot e(\beta', Q)^{-1} = e(\mathbf{d}\mathbf{e}^t, c_1) \cdot e(\beta, Q)^{-1} = e(\chi, P)
\end{aligned}$$

Since the group G_T is a prime order group, using the bi-linearity of e, we get $\pi' = \pi$. Since $(\rho', \hat{\rho}', \mathtt{lbl}') = (\rho, \hat{\rho}, \mathtt{lbl})$ as well, we get a contradiction.

□

C Appendix: Key Exchange in the PAK Model

C.1 PAK Model of Security

We describe a definition of security, which we refer to as PAK model in this paper, due to Bellare, Pointcheval and Rogaway [1]. This description summarizes a version depicted in [15].

Participants, passwords, and initialization. Prior to any execution of the protocol there is an initialization phase during which public parameters are established. We assume a fixed set User of protocol participants. For every distinct $U, U' \in \mathsf{User}$, we assume U and U' share a password $pw_{U,U'}$. We make the simplifying assumption that each $pw_{U,U'}$ is chosen independently and uniformly randomly from the set $[D] \stackrel{\text{def}}{=} \{1, \ldots, D\}$ for some integer D.

Execution of the protocol. We denote instance i of user U as Π_U^i.

- $\mathsf{sid}_U^i, \mathsf{pid}_U^i$, and sk_U^i denote the *session key*, *partner id* and *session key* for an instance, respectively.
- acc_U^i and term_U^i are boolean variables denoting whether a given instance has accepted or terminated, respectively.

The adversary's interaction with the principals (more specifically, with the various instances) is modeled via access to *oracles* which we describe now:

- $\mathsf{Send}(U, i, \mathsf{msg})$ — This sends message msg to instance Π_U^i. This instance runs according to the protocol specification. The output of Π_U^i is given to the adversary.

  The adversary can "prompt" instance Π_U^i to initiate the protocol with partner U' by query $\mathsf{Send}(U, i, U')$. In response to this query, instance Π_U^i outputs the first message of the protocol.

- $\mathsf{Execute}(U, i, U', j)$ — If Π_U^i and $\Pi_{U'}^j$ have not yet been used, this oracle executes the protocol between these instances and gives the transcript of this execution to the adversary. This oracle call represents passive eavesdropping of a protocol execution.

- $\mathsf{Reveal}(U, i)$ — This outputs the session key sk_U^i, modeling leakage of session keys.

- $\mathsf{Test}(U, i)$ — A random bit b is chosen; if $b = 1$ the adversary is given sk_U^i, and if $b = 0$ the adversary is given a session key chosen uniformly from the appropriate space.

Partnering. Let $U, U' \in \mathsf{User}$. Instances Π_U^i and $\Pi_{U'}^j$ are *partnered* if: (1) $\mathsf{sid}_U^i = \mathsf{sid}_{U'}^j \neq \text{NULL}$; and (2) $\mathsf{pid}_U^i = U'$ and $\mathsf{pid}_{U'}^j = U$.

Correctness. If Π_U^i and $\Pi_{U'}^j$ are partnered then $\mathsf{acc}_U^i = \mathsf{acc}_{U'}^j = \text{TRUE}$ and $\mathsf{sk}_U^i = \mathsf{sk}_{U'}^j$.

Advantage of the adversary. An instance Π_U^i is *fresh* unless one of the following is true at the conclusion of the experiment:

1. at some point, the adversary queried $\mathsf{Reveal}(U, i)$;
2. or, at some point, the adversary queried $\mathsf{Reveal}(U', j)$, where $\Pi_{U'}^j$ and Π_U^i are partnered.

An adversary $\mathcal{A}$ *succeeds* if it makes a single query $\mathsf{Test}(U, i)$ to a fresh instance Π_U^i, and outputs a bit b' with $b' = b$. We denote this event by Succ. The *advantage* of the adversary $\mathcal{A}$ in attacking Π is given by $\text{ADV}_{\mathcal{A},\Pi}(k) \stackrel{\text{def}}{=} 2 \cdot \Pr[\mathsf{Succ}] - 1$, where the probability is taken over the random coins used during the course of the experiment (including the initialization phase).

Definition 8 *Protocol* Π *is a* secure protocol for password-based authenticated key exchange *if, for all dictionary sizes* D *and for all* PPT *adversaries* $\mathcal{A}$ *making at most* $Q(n)$ *on-line attacks, it holds that* $\text{ADV}_{\mathcal{A},\Pi}(n) \leq Q(n)/D + \mathsf{negl}(n)$.

C.2 Proof of Security of the PAK protocol

Theorem 9 ([16]) *The the protocol in Figure 1 is secure in the PAK model provided that the encryption scheme is CCA2 secure and the hash function is smooth projective with respect to the language of labeled encryption of the password.*

Proof. We construct a simulator for the protocol execution which information theoretically hides the password in messages sent to the adversary and the keys generated. Hence the only way for the adversary to influence the key generation non-trivially is to demonstrate knowlege of the password pwd.

The secret key sk for the CCA2-secure encryption scheme is given to the simulator. The main difference in the simulation from the protocol is that S uses a dummy message μ instead of the real password which it does not have access to.

C.2.1 Passive Execute Queries

When the adversary asks for the transcript of a session between parties P_i and P_j, the simulator uses encryptions of μ, instead of the password:

$$k_i \xleftarrow{\$} \text{Hash-}K;\ s_i \leftarrow \alpha(k_i) \tag{7}$$

$$label_i \leftarrow (P_i, P_j, s_i) \tag{8}$$

$$C_i \xleftarrow{\$} \mathsf{enc}_{pk}(label_i, \mu) \tag{9}$$

$$\texttt{Send}\ \ (label_i, C_i) \tag{10}$$

Similarly for the party P_j. The key of both parties for this session is set to a common random value. We show that this transcript is indistinguishable from the real protocol by outlining a sequence of games starting from the real transcript and ending with the simulated transcript. Suppose P_j generates the message $(label_j, C_j) = (P_j, P_i, s_j, C_j)$.

Game 0: This is just the real protocol. We note that P_i computes its key as follows: $sk_i \leftarrow privH_{k_i}(label_j, C_j, \text{pwd}) \cdot pubH_{s_j}(label_i, C_i, \text{pwd}; x_i)$. The private hash algorithm $privH$ uses the private hash key for P_i, whereas the public hash algorithm $pubH$ uses the public hash key received and the witness for the encryption produced by P_i, that is, x_i.

Game 1: This is the real protocol with the following change. Since s_j is the public hash key actually generated by a session of P_j, the simulator knows the corresponding private hash key k_j. In this game, P_i computes its key using private hash keys only. That is, $sk_i \leftarrow privH_{k_i}(label_j, C_j, \text{pwd}) \cdot privH_{k_j}(label_i, C_i, \text{pwd})$. The transcript remains identical to Game 0.

Game 2: This is Game 1 with the following change. C_i and C_j are both computed as encryptions of μ. This is indistinguishable from Game 1 due to CCA2 security of the encryption scheme and because the encryption randomness is not used anywhere.

Game 3: This is Game 2 with the following change. The key of both parties for this session is set to a common random value. In Game 2 the key computated by both peers was $privH_{k_i}(label_j, C_j, \text{pwd}) \cdot privH_{k_j}(label_i, C_i, \text{pwd})$. Since both the ciphertexts are encryptions of μ, the hashes are over non-members of the language associated with it. Also the private hash keys k_i and k_j are not used by any

other sessions of any principal. Hence the generated key is distributed uniformly from random. Therefore Game 3 is indistinguishable from Game 2.

Since Game 3 is just the simulated transcript we are done with this part.

C.2.2 New Session: Sending a message to $\mathcal{A}$

Next we look at the active attack queries. S starts simulating a new session of the protocol Π for party P_i, peer P_j, and CRS $= pk$. To simulate this session, S uses encryptions of μ, instead of the password:

$$k_i \xleftarrow{\$} \text{Hash-}K;\ s_i \leftarrow \alpha(k_i) \tag{11}$$

$$label_i \leftarrow (P_i, P_j, s_i) \tag{12}$$

$$C_i \xleftarrow{\$} \mathsf{enc}_{pk}(label_i, \mu) \tag{13}$$

$$\texttt{Send}\ \ (label_i, C_i) \tag{14}$$

C.2.3 On Receiving a Message from $\mathcal{A}$

On receiving a message $(label'_j, C'_j) = (P_j, P_i, s'_j, C'_j)$ from $\mathcal{A}$ purportedly from P_j, intended for this session of P_i, the simulator S first public verifies C'_j to be a ciphertext with label (P_j, P_i, s'_j). If the checks fail, the session is aborted.

We call a tuple (P_j, P_i, s, C) *well formed* for this session of P_i, if C can be public verified to be a valid ciphertext with label $label$. We call a well formed tuple to be *legitimate* if there is an actual session of P_j which generated this message. Two sessions of principals P_i and P_j are called partnered if P_i and P_j are peers in both sessions, and if the session of P_i receives the message produced by the session of P_j and vice versa.

If (P_j, P_i, s'_j, C'_j) is legitimate for this session and this is a partnered session with peer P_j, then choose a key value uniformly and independently from random and set it as the key for both the sessions. If there is no partner for this session, then set its key to be uniformly and independently random.

If the message is not even well formed then, as in the real protocol, the session is terminated. However, if the message is well formed but not legitimate for this session, then the simulator behavior is as follows. Let p' be the decryption of C'_j with label (P_j, P_i, s'_j). If $p' = \text{pwd}$, then declare A to be successful and terminate. Otherwise set sk_i to be random.

Since the messages sent out are information theoretically independent of the password, the probability of the decryption equaling the password is

at most the probability of guessing the password by the adversary. In the next section, we describe a sequence of games leading from the real protocol finally ending up at the simulator, which will complete the proof.

C.2.4 Proof of Indistinguishability for the simulator

Now we outline a sequence of games leading from the real protocol, which we denote $\mathbf{Expt}_0$ finally ending up at the simulator described above.

$\mathbf{Expt}_1$: In this experiment, just as in the real protocol, P_i computes the first message as follows: generate private hash key k_i and let $s_i \leftarrow \alpha(k_i)$ be the public hash key. Generate randomness x_i and compute an encryption of pwd with label $label_i = (P_i, P_j, s_i)$. That is, compute $C_i = \mathsf{enc}_{pk}(label_i, \text{pwd}; x_i)$. Send $(label_i, C_i)$ to the adversary A.

On receiving a message (P_j, P_i, s'_j, C'_j) from the adversary A in this session, just as in the real protocol, the key sk_i is set to the actual computed key in case the message is legitimate for this session. That is, $sk_i \leftarrow privH_{k_i}(label'_j, C'_j, \text{pwd}) \cdot pubH_{s'_j}(label_i, C_i, \text{pwd}; x_i)$. The private hash algorithm $privH$ uses the private hash key for P_i, whereas the public hash algorithm $pubH$ uses the public hash key received and the witness for the encryption produced by P_i, that is, x_i.

If the message is not even well formed then, as in the real protocol, the session is terminated. However, if the message is well formed but not legitimate for this session, then the behavior is different and as follows. Let p' be the decryption of C'_j with label (P_j, P_i, s'_j). If $p' = \text{pwd}$, then declare A to be successful and terminate. Otherwise set sk_i to be random.

In the case of $p' \neq \text{pwd}$, we have that $(label'_j, C'_j, \text{pwd})$ is not in the language of the smooth hash. Also note that the private hash key k_i is not used anywhere other than this session. Hence the private hash $privH_{k_i}(label'_j, C'_j, \text{pwd})$ is uniformly random, even conditioned on $s_i = \alpha(k_i)$. Hence this step is indistinguishable to A from the real key computation. Making A succeed if $p' = \text{pwd}$ can only increase the advantage of A.

$\mathbf{Expt}_2$: This experiment is the same as $\mathbf{Expt}_1$ with the following change. On receiving a message (P_j, P_i, s'_j, C'_j) from the adversary A in this session, the key sk_i is computed in a different way in case the message is legitimate for this session. Since the message is legitimate, the simulator knows the private hash key k'_j corresponding to s'_j. It now computes the second hash with the private key rather than the public key. That is,

$sk_i \leftarrow privH_{k_i}(label'_j, C'_j, \text{pwd}) \cdot privH_{k'_j}(label_i, C_i, \text{pwd})$. Since the message was legitimate, the hash is over a language member and hence the computations are actually the same. Observe that now the encryption randomness x_i is not used anywhere.

$\mathbf{Expt}_3$: This experiment is the same as $\mathbf{Expt}_2$ with the following changes. P_i sends out a message where the password pwd has been substituted by μ: $C_i \leftarrow \mathsf{enc}_{pk}(label_i, \mu)$. The key generation is changed in this experiment as follows. For partnered sessions, a key is generated uniformly from random and is set as the key for both the sessions. For all other sessions which receive legitimate messages, the key is generated uniformly and independently from random. The experiment remains unchanged for sessions which do not receive legitimate messages.

The indistinguishability of Experiments 2 and 3 follow by a sequence of hybrid experiments, where we change, session by session, the message being sent out from using the password pwd to using μ. Consider the sessions in the order of sending the first message. In this order, the i-th session is changed as follows, resulting in $\mathbf{Expt}_{2,i}$. Suppose the session is of principal P and its peer is Q. We change the message being sent by P from $(label_i, C_i)$ to $(label_i, \hat{C}_i)$, where $\hat{C}_i = \mathsf{enc}_{pk}(label_i, \mu)$.

Consider the scenario where this session receives a legitimate message (P_j, P_i, s'_j, C'_j) - a message generated by the j-th session in the order. We have two cases depending on whether i is less than or greater than j. Consider the case where i is less than j. In this case, set the key of session i to be uniformly and independently random instead of being computed by the projective hashes. If session j receives $(P_i, P_j, s_i, \hat{C}_i)$, then set the key of session j to be the same as session i. The key of all other sessions which receive $(P_i, P_j, s_i, \hat{C}_i)$, and are greater than i in the order, are set to be uniformly and independently random. Consider the case where i is greater than j. In this case, the key of session i was already set when the session j was considered - we just retain that. The key of all other sessions which receive $(s_i, \hat{C}_i)$, and are greater than i in the order, are set to be uniformly and independently random.

We show that $\mathbf{Expt}_{2,i}$ and $\mathbf{Expt}_{2,i-1}$ are indistinguishable. The ciphertexts C_i and $\hat{C}_i$ are indistinguishable by the CCA2 security of the encryption scheme and the fact that the encryption randomness is not used in any other computation. If session j is the partnered session of session i and j is greater than i, then in $\mathbf{Expt}_{2,i}$ they both compute the key as $privH_{k_i}(label_j, C_j, \text{pwd}) \cdot privH_{k_j}(label_i, \hat{C}_i, \text{pwd})$. Observe firstly,

$(label_i, \hat{C}_i, \text{pwd})$ is not in the language of the hash. Secondly, sessions other than i which use the private key k_j for computing the second hash in $\mathbf{Expt}_{2,i}$, are all greater in order than i. Hence they compute hash over language members only. Due to these two observations, sessions i and j compute the same key, which is uniformly and independently random with respect to all other sessions. If the partnered session j is less than i in the order, then the key computation remains unchanged.

Now consider a session $l \neq j$ greater than i in the order, which receives $(P_i, P_j, s_i, \hat{C}_i)$. In $\mathbf{Expt}_{2,i}$ it computes the key as $privH_{k_i}(label_l, C_l, \text{pwd}) \cdot privH_{k_l}(label_i, \hat{C}_i, \text{pwd})$. Again we have, $(label_i, \hat{C}_i, \text{pwd})$ is not in the language of the hash. Again, sessions which use the private key k_l for computing the second hash in $\mathbf{Expt}_{2,i}$, are all greater in order than i. Hence they compute hash over language members only. Due to these two observations, session l computes a key, which is uniformly and independently random with respect to all other sessions. If l is less than i in the order, then l's key computation remains unchanged.

We end up at $\mathbf{Expt}_3$ after all these hybrid experiments, and hence it is indistinguishable from $\mathbf{Expt}_2$. Since $\mathbf{Expt}_3$ is just the simulator we are through.

D More Efficient Unbounded Simulation Sound NIZKs

In [4], an unbounded simulation sound NIZK scheme is given for bilinear groups, building on the Groth-Sahai NIZKs and using Cramer-Shoup like CCA2 encryption schemes under K-linear assumptions. In this section we show various general optimizations for that construction, and further optimizations for specific languages involving generalized Diffie-Hellman tuples.

The general optimizations can be summarized as follows.

1. The scheme in [4] uses a one-time signature scheme. However, since it also uses a labeled CCA2 encryption scheme, the one-time signature scheme can be dropped, and one can use the label in the CCA2 scheme to get the signature property.

2. The scheme in [4] allows the simulator to generate a CCA2 encryption of u^x (for trapdoor x) along with a proof, instead of the proof of the statement. In order for the Adversary to cheat, it must also produce such an encryption, which is impossible under CCA2. However,

one notices that since the simulator knows u^x, instead of a normal encryption, the simulator can hide u^x with just the smooth hash.

We now give this optimized version under the SXDH-assumption (for ease of exposition). Similar optimizations can be obtained under the DLIN assumption. Let the SXDH-group be (G_1, G_2, G_T), each of the groups of order q, with a $\mathbb{Z}_q$-bilinear map e from $G_1 \times G_2$ to G_T. We will write the bilinear map $e(A, B)$ in infix notation as $A \cdot B$. The group operation will be written in additive notation.

Languages for the simulation-sound NIZK can be specified by equations (relations) of the form $\vec{x} \cdot \vec{A} = T$, where $\vec{x}$ are variables from $\mathbb{Z}_q$, $\vec{A}$ are constants from G_2, and T is a constant from G_T, or vice-versa, and thus $\vec{x}$ serves as witness for a member of a language specified by $\vec{A}$ and T. Languages can also be specified by equations of the form $\vec{B} \cdot \vec{\mathcal{Y}} = T_1 \cdot T_2$, where $\vec{B}$ are elements from G_1, $\vec{\mathcal{Y}}$ are variables from G_2, and T_1 and T_2 are constants from G_1 and G_2 resp. One can also consider languages with multiple such relations of both kinds.

Note that languages for which Groth-Sahai NIWI proofs can be given are more general, including equations like $\vec{x} \cdot \vec{A} + \vec{b} \cdot \vec{\mathcal{Y}} = T$, as well as quadratic equations.

The uss-NIZK CRS will consist of the usual Groth-Sahai NIWI CRS for SXDH, along with $g, A{=}g^a, \mathbf{k} = g^{k_1} A^{k_2}, \mathbf{d}{=}g^{d_1} A^{d_2}, \mathbf{e}{=}g^{e_1} A^{e_2}$, and $\mathbf{h}{=}g^x, \mathbf{u}{=}g^u$, with $g \in G_1$, and $a, k_1, k_2, d_1, d_2, e_1, e_2, x, u$ chosen at random from $\mathbb{Z}_q$. One could alternatively choose these values from G_2. Let H be a collision resistant hash function.

Given a set of relations as above, along with satisfying variables, the prover does the following:

1.
   - For each equation of the kind $\vec{x} \cdot \vec{A} = T$, it generates a modified equation $\vec{x} \cdot \vec{A} = \delta \cdot T$, where δ is a new global integer variable.
   - Get modified equations of the form $\vec{B} \cdot \vec{\mathcal{Y}} + T_1 \cdot \mathcal{V} = 0$, where $\mathcal{V}$ is a new variable representing elements from G_2, along with an additional equation $\mathcal{V} - \delta \cdot T_2 = 0$ [13].
   - Generate an additional quadratic equation $\delta(1 - \delta) = 0$.

2. Produce a Groth-Sahai NIWI proof for the above modified set of equations, with δ set to 1. Call this proof, which includes all commitments to original variables as well as δ and $\mathcal{V}$, as π_1. Also append the original statement to be proven in π_1.

3. Generate $\rho = g^w$, $\hat{\rho} = A^w$, with w chosen at random.

4. Produce a Groth-Sahai NIWI proof of the following statements (using the same commitment to δ as in step 2, and w', x' committed to zero): $\rho^{1-\delta} = g^{w'}$, $\hat{\rho}^{1-\delta} = A^{w'}$, $\mathbf{h}^{1-\delta} = g^{x'}$. Call this proof along with commitments to x', w' as π_2.

5. Set $b = \mathbf{u} \cdot (\mathbf{kde}^t)^w$, where $t = H(\rho, \hat{\rho}, \pi_1, \pi_2)$.

6. Produce a Groth-Sahai NIWI proof of the following statement (using the same commitment to δ as in step 2, and same commitment for w', x' as in Step 4): $b^{1-\delta} = \mathbf{u}^{x'} \cdot (\mathbf{kde}^t)^{w'}$. Call this proof π_3.

7. The uss-NIZK proof consists of $(\pi_1, \pi_2, \pi_3, \rho, \hat{\rho}, b)$.

The proof of zero-knowledge is similar to the construction in [4]. The proof of unbounded simulation sound extractability is also similar to as in [4] but using the CCA2 encryption scheme (and its proof) as described in Section 5.

It is noteworthy that the uss-NIZK CRS can just give the product of $\mathbf{k}$ and $\mathbf{d}$, and it follows that $\mathbf{k}$ can be deleted altogether from the scheme. The above can also be made a labeled unbounded simulation-sound extractable NIZK, by including the label in the collision-resistance hash computation t in step 5.

Note that it takes 14 extra group elements to convert a NIZK proof into a uSS-proof using this construction. This follows from the description in [13] for the SXDH construction. In the case of DLIN assumption, one would need 28 extra group elements. For the language in Section 8.3, the NIZK proof requires 18 group elements. In the full paper [14] we show a further optimization for this specific language, which saves another 3 group elements, resulting in a total of 14+18-3 = 29 group elements for the uss-NIZK proof for the language.

E Secure Protocols under DLIN Assumption

In this section, we instantiate the protocols under the DLIN assumption. Let G be a group with a bilinear pairing $e : G \times G \to G_T$ and $|G| = |G_T| = q$, a prime number. Also assume that DLIN is hard for G. Let $L_{g,f,h}$ be the language: $\{(\rho, \sigma, \tau) \in G^3 \mid \exists x, y.\ \rho = g^x \wedge \sigma = f^y \wedge \tau = h^{x+y}\}$, with g, f, h in G. The proofs are analogous to the SXDH versions and these generalizations can be obtained as in [7].

E.1 Single Theorem Relatively-Sound NIZK for the DLIN Language

We construct a single-theorem NIZK proof system, with a private verification function for $L_{g,f,h}$, which is relatively-sound, as follows:

CRS Generation: Generate $d_1, d_2, e_1, e_2, u_1, u_2 \xleftarrow{\$} \mathbb{Z}_p$ and $\tilde{\psi}$, a CRS for a Groth-Sahai NIWI under the DLIN assumption. Compute $(\mathbf{d}_1, \mathbf{e}_1, \mathbf{d}_2, \mathbf{e}_2) = (g^{d_1}h^{u_1}, f^{e_1}h^{u_1}, g^{d_2}h^{u_2}, f^{e_2}h^{u_2})$. The CRS is $\psi = (\tilde{\psi}, \mathbf{d}_1, \mathbf{e}_1, \mathbf{d}_2, \mathbf{e}_2)$. The last four elements are the projection keys for a 2-universal projective-hash for the DLIN language (just as [7]), to be used in the relatively simulation sound system. The private verification trapdoor key is $\xi = (d_1, d_2, e_1, e_2, u_1, u_2)$.

Prover: Given witness x, y and candidate (g^x, f^y, h^{x+y}), construct proof as follows. Let *com* be Groth-Sahai commitments to exponents x, y. Compute $t \leftarrow H(g^x, f^y, h^{x+y}, com)$, where H is a collision resistant hash function. Then compute $\beta \leftarrow (\mathbf{d}_1{\mathbf{e}_1}^t)^x(\mathbf{d}_2{\mathbf{e}_2}^t)^y$. This is a 2-universal projective-hash computed on the candidate with witness x, y. Let $\tilde{\pi}$ be the Groth-Sahai NIWI proof (which also happens to be a NIZK proof) for the language $\{\rho, \sigma, \tau, \beta \mid \exists x, y \ : \rho = g^x, \sigma = f^y, \tau = h^{x+y}, \beta = (\mathbf{d}_1{\mathbf{e}_1}^t)^x(\mathbf{d}_2{\mathbf{e}_2}^t)^y$, where t is a hash of ρ, σ, τ, *and the commitment* to x, y in the NIWI itself. Output proof $\pi = (\beta, \tilde{\pi})$.

Public Verify: Given $\pi = (\beta, \tilde{\pi})$ as a candidate proof of (ρ, σ, τ), let *com* be the witness commitments part of $\tilde{\pi}$. Compute $t \leftarrow H(\rho, \sigma, \tau, com)$. Then check $\tilde{\pi}$ as a Groth-Sahai NIWI proof for the statement $\exists x, y \ : \rho = g^x, \sigma = f^y, \tau = h^{x+y}, \beta = (\mathbf{d}_1{\mathbf{e}_1}^t)^x(\mathbf{d}_2{\mathbf{e}_2}^t)^y$

Private Verify: Given $\pi = (\beta, \tilde{\pi})$ as a candidate proof of (ρ, σ, τ), let *com* be the witness commitments part of $\tilde{\pi}$. Compute $t \leftarrow H(\rho, \sigma, \tau, com)$. Then first do public verification and if that succeeds then check the following equation: $\beta \stackrel{?}{=} \left(\rho^{d_1}\sigma^{e_1}\tau^{u_1}\right)\left(\rho^{d_2}\sigma^{e_2}\tau^{u_2}\right)^t$.

Theorem 10 *The above system is a single-theorem relatively-sound NIZK proof system for $L_{g,f,h}$.*

Again, it is worth pointing out here that we use the fact that in Groth-Sahai NIZKs for DLIN, once the commitments to the witnesses are fixed, there is a unique proof satisfying the linear equations of the type used in the above NIZK proof.

Again, it is easy to extend this proof system to a labeled version in the following way. Compute t as before, but additionally include the label. That is, compute t as $H(\rho, \sigma, \tau, com, label)$.

E.2 Public Verifiable CCA2 Encryption

We now define a *labeled* publicly-verifiable public-key encryption scheme DLENC as follows:

Key Generation: Generate $g, f, h \xleftarrow{\$} \mathcal{G}$, and $k_1, k_2 \xleftarrow{\$} \mathbb{Z}_q$. Let $K_1 = g^{k_1}$ and $K_2 = f^{k_2}$. Let ψ be the CRS for an l-SRS-NIZK for the language $L_{g,f,h}$. The public key is $(g, f, h, K_1, K_2, \psi)$ and the private key is (k_1, k_2).

Encrypt: Given plaintext $m \in \mathcal{G}$, and label `lbl`. Choose $x, y \xleftarrow{\$} \mathbb{Z}_q$. Let the tuple $\langle \rho, \sigma, \tau, \gamma \rangle$ be $\langle g^x, f^y, h^{x+y}, m \cdot {K_1}^x {K_2}^y \rangle$. Let π be an l-SRS-NIZK proof of $(\rho, \sigma, \tau) \in L_{g,f,h}$ with witness (x, y) and label $(\gamma, \texttt{lbl})$. The ciphertext is $(\rho, \sigma, \tau, \gamma, \pi)$.

Decrypt: Given ciphertext $c = (\rho, \sigma, \tau, \gamma, \pi)$ and label `lbl`. Verify if π is an l-SRS-NIZK proof for (ρ, σ, τ) and label $(\gamma, \texttt{lbl})$. If verification fails output $\perp$. Otherwise output $m = \frac{\gamma}{\rho^{k_1} \sigma^{k_2}}$.

Verify: Given ciphertext $c = (\rho, \sigma, \tau, \gamma, \pi)$ and label `lbl`. Verify if π is an l-SRS-NIZK proof for (ρ, σ, τ) and label $(\gamma, \texttt{lbl})$. If verification fails output false else output true.

Theorem 11 *The scheme* DLENC *is publicly-verifiable (labeled) IND-CCA2 secure under the DLIN assumption.*

E.3 Secure Protocol in the PAK Model

We again instantiate the [16] scheme, but now under the DLIN assumption. The public verifiable encryption is the scheme DLENC as described before. Let the public key for the DLENC scheme be $(g, f, h, K_1, K_2, \psi)$. The hash proof system is described as follows:

Key Generation: Generate $l, m, n \xleftarrow{\$} \mathbb{Z}_q$. Compute $\eta \leftarrow g^l (K_1)^n$ and $\phi \leftarrow f^m (K_2)^n$. The public key is (η, ϕ) and the private key is (l, m, n).

Public Hash Computation: Given parameters $(label, c, msg)$, where $c = \langle \rho, \sigma, \tau, \gamma, \pi \rangle$. Also given $(\rho, \sigma, \tau; x, y) \in \mathcal{R}_{f,g,h}$. Then compute hash as:

$$H_{\eta,\phi}(label, c, msg) \leftarrow \eta^x \phi^y$$

Private Hash Computation: Given parameters $(label, c, msg)$, where $c = \langle \rho, \sigma, \tau, \gamma, \pi \rangle$. Then compute hash as:

$$H_{l,m,n}(label, c, msg) \leftarrow \rho^l \sigma^m \left(\frac{\gamma}{msg}\right)^n$$

Theorem 12 *Assume the existence of a DLIN hard group G which supports a bilinear pairing operation. Then the protocol in Figure 1 with encryption instantiated by* DLENC *and hash proof system instantiated as described, is secure in the PAK model.*

E.4 Secure PWKE-Protocol in the UC/DLIN Model

In Figure **??**, we give a UC-secure PWKE-protocol under the Decisional Linear assumption (DLIN).

Theorem 13 *Assume the existence of a DLIN-hard group, a labeled unbounded simulation-sound G-extractable NIZK proof system. Then the protocol in Figure* **??** *securely realizes the* $\widehat{\mathcal{F}}_{\text{PWKE}}$ *functionality in the* $\mathcal{F}_{\mathsf{crs}}$ *hybrid model, in the presence of static corruption adversaries.*

$\text{CRS} = g, f, h, K_1, K_2, \mathcal{P}, \psi : \quad g, f, h, K_1, K_2 \xleftarrow{\$} G \quad \mathcal{P} \xleftarrow{\$} G \quad \psi = \text{uSS-NIZK } \mathsf{CRS}$

Party P_i		Adversary $\mathcal{A}$
Input $(\texttt{NewSession}, sid, \mathsf{ssid}, P_i, P_j, \text{pwd}, initiator/responder)$		
Choose $x_i, y_i, l_i, m_i, n_i \xleftarrow{\$} \mathbb{Z}_q^*$.		
Set $\begin{bmatrix} \rho_i = g^{x_i}, \sigma_i = f^{y_i}, \tau = h^{x_i+y_i}, \\ \gamma_i = \text{pwd} \cdot (K_1^{x_i} K_2^{y_i})^{\mathsf{ssid}}, \\ \eta_i = g^{l_i} K_1^{n_i}, \phi_i = f^{m_i} K_2^{n_i} \end{bmatrix}$	$\xrightarrow{c_i, \eta_i, \phi_i, \pi_i}$	$\mathcal{A}$
Let $c_i = \langle \rho_i, \sigma_i, \tau_i, \gamma_i \rangle$, and		
$\pi_i = \text{uSS-NIZK}_\psi \left(\begin{matrix} \rho_i, \sigma_i, \tau_i, \eta_i, \phi_i; \\ x_i, y_i, \mathcal{P}^{l_i}, \mathcal{P}^{m_i}, \mathcal{P}^{n_i} \end{matrix} \right)$ with label $\langle P_i, P_j, \mathsf{ssid} \rangle$.		
	$\xleftarrow{c'_j, \eta'_j, \phi'_j, \pi'_j}$	$\mathcal{A}$
Let $c'_j = \langle \rho'_j, \sigma'_j, \tau'_j, \gamma'_j \rangle$.		
If any of $\rho'_j, \sigma'_j, \tau'_j, \gamma'_j, \eta'_j, \phi'_j$ is not in $G \backslash \{1\}$, or		
not uSS-NIZK-Verify$(\pi'_j; \rho'_j, \sigma'_j, \tau'_j, \eta'_j, \phi'_j)$ with label $\langle P_j, P_i, \mathsf{ssid} \rangle$		
Set $\text{sk}_i \xleftarrow{\$} G_T$,		
else compute $\begin{bmatrix} h'_2 = (\rho'_j)^{l_i} (\sigma'_j)^{m_i} (\frac{\gamma'_j}{\text{pwd}})^{n_i} \\ h_1 = (\eta'_j)^{x_i} (\phi'_j)^{y_i} \end{bmatrix}$		
Set $\text{sk}_i = e(h'_2 \cdot h_1, \mathcal{P})$.		
Output $(sid, \mathsf{ssid}, \text{sk}_i)$.		

Single round UC-secure Password-based Authenticated Key Exchange in DLIN Bilinear group G of prime order q. It assumes a group G with a $\mathbb{Z}_q$-bilinear map e from $G \times G$ to G_T. Let g and $\mathcal{P}$ be generators of G. The shared password pwd is assumed to be in G, and ssid is assumed to be in $\mathbb{Z}_q$. The uSS-NIZK$(\rho, \sigma, \tau, \eta, \phi;\ x, y, L, M, N)$ is a labeled unbounded-simulation G-extractable NIZK proof of membership in the language $L = \{\rho, \sigma, \tau, \eta, \phi \mid \exists x, y, L, M, N : \rho = g^x,\ \sigma = f^y,\ \tau = h^{x+y},\ e(\eta, \mathcal{P}) = e(g, L) e(K_1, N),\ e(\phi, \mathcal{P}) = e(f, M) e(K_2, N)\ \}$

Figure 4: UC Protocol for Password-based Key Exchange in DLIN Group G