Round-Optimal Password-Based Authenticated Key Exchange

Jonathan Katz[1,*] and Vinod Vaikuntanathan[2,**]

[1] University of Maryland, USA
jkatz@cs.umd.edu
[2] Microsoft Research
vinodv@alum.mit.edu

Abstract. We show a general framework for constructing password-based authenticated key exchange protocols with *optimal* round complexity — one message per party, sent simultaneously — in the standard model, assuming a common reference string. When our framework is instantiated using bilinear-map cryptosystems, the resulting protocol is also (reasonably) efficient. Somewhat surprisingly, our framework can be adapted to give protocols in the standard model that are *universally composable* while still using only one (simultaneous) round.

1 Password-Based Authenticated Key Exchange

Protocols for *authenticated key exchange* enable two parties to generate a shared, cryptographically strong key while communicating over an insecure network under the complete control of an adversary. Such protocols are among the most widely used and fundamental cryptographic primitives; indeed, agreement on a shared key is necessary before "higher-level" tasks such as encryption and message authentication become possible.

Parties must share *some* information in order for authenticated key exchange to be possible. It is well known that shared cryptographic keys — either in the form of public keys or a long, uniformly random symmetric key — suffice, and several protocols in this model, building on the classic Diffie-Hellman protocol [16] (which protects only against an eavesdropping adversary and provides no authentication at all) are known; see, e.g., [7,4].

Password-based protocols allow users to "bootstrap" even a *very weak* (e.g., short) shared secret into a (much longer) cryptographic key. The canonical application here is authentication using *passwords*, though protocols developed in this context can be useful even when the shared secret has high min-entropy (but is not uniform) [9]. The security guaranteed by password-based protocols (roughly speaking) is that if the password is chosen uniformly[1] from a dictionary

[*] Work done in part while visiting IBM. Research supported by NSF grant #0627306 and NSF CAREER award #0447075.

[**] Work done while at IBM.

[1] Although the usual presentation of PAK assumes a uniform password, known protocols work with passwords chosen from any (efficiently sampleable) distribution.

of size D then an adversary who initiates Q "on-line" attacks — i.e., who actively interferes in Q sessions — has "advantage" at most Q/D. (This is inherent, as an adversary can always carry out Q impersonation attempts and succeed with this probability.) In particular, "off-line" dictionary attacks where an adversary enumerates passwords from the dictionary of potential passwords, and tries to match observed protocol executions to each one, are of no use.

Early work [20, 24] considered a "hybrid" setting where users share public keys in addition to a password. In the setting where *only* a password is shared, Bellovin and Merritt [6] proposed the first protocols for password-based authenticated key exchange (PAK) with heuristic arguments for their security. Several years later, provably secure PAK protocols were constructed [3,10,31] in the random oracle/ideal cipher models, and many improvements and generalizations of these protocols are known. In contrast, only a handful of PAK protocols are known in the so-called "standard model" (i.e., without random oracles):

– **General assumptions:** Goldreich and Lindell [19] gave the first PAK protocol in the standard model. Subsequent work of Barak et al. [2] shows a general feasibility result for computation over unauthenticated networks which implies a solution for PAK as a special case. These approaches gives the only PAK protocols for the plain model where there is no setup. (Nguyen and Vadhan [33] show efficiency improvements to the Goldreich-Lindell protocol, but achieve a weaker notion of security.) These approaches are impractical in terms of communication, computation, and round complexity. Moreover, they do not tolerate concurrent executions by the same party (unless additional setup is assumed). A recent protocol of Goyal et al. [21] addresses the issue of concurrent executions, but is still far from practical.
– **Efficient protocols:** Katz, Ostrovsky, and Yung [28] demonstrated the first *efficient* PAK protocol with a proof of security based on standard assumptions; extensions and improvements of their protocol were given in [18,13,27, 17,30]. Different constructions of efficient PAK protocols are given in [26,22]. In contrast to the works mentioned earlier, these approaches are secure even under concurrent executions by the same party. On the other hand, they require a *common reference string* (CRS). In practice, however, a CRS does not appear to be a serious drawback in the context of PAK where the CRS can be hard-coded into an implementation of the protocol. We note also that reliance on a CRS (or some other setup) is inherent for achieving *universally composable* PAK [13].

Round/message complexity of existing protocols. We distinguish between *rounds* and *messages*. Differing somewhat from the usual convention in the two-party setting (but matching the usual convention in the multi-party setting), we let a round consist of one message sent by each party simultaneously; note that in a one-round protocol each honest party's message (if any) cannot depend on the other party's message. We stress, however, that even for one-round protocols the adversary is always assumed to be *rushing*; i.e., the adversary may wait to receive an honest party's first-round message before sending its own.

Determining the optimal round complexity of key-exchange protocols is of both theoretical and practical interest, and has been studied in various settings. The original Diffie-Hellman protocol [16], which provides security against a passive eavesdropper, can be run in one round; one-round authenticated key exchange based on shared public/symmetric keys is also possible [25, 34]. One-round protocols for PAK are also known (e.g., [3]) in the random oracle model. All prior PAK protocols based on standard assumptions, though, require three or more rounds. We remark that the protocols in [26, 22] achieve *explicit* authentication in three rounds (whereas the protocols of [28,18,17,30] achieve only *implicit* authentication in three rounds, and require an additional round for explicit authentication), but the round complexity of these protocols cannot be reduced even if only implicit authentication is desired.

1.1 Our Results

We show a new framework for constructing *one-round* PAK protocols in the standard model (assuming a CRS), where each party may send their message *simultaneously*. (Once again, we stress that our security model allows for a "rushing" adversary who waits to see the message sent by a party before sending its response.) Our protocols achieve implicit authentication but can be extended to give explicit authentication using one additional round; it is not hard to see that explicit authentication is impossible in one round without stronger setup assumptions (e.g., a global clock).

Our framework relies on non-interactive zero-knowledge proofs (NIZK) and so, in general, may be computationally inefficient. When instantiating our framework using bilinear maps, however, we obtain a reasonably efficient solution (e.g., communicating a constant number of group elements).

Somewhat surprisingly, we can extend our framework to give a universally composable PAK protocol [12] *without increasing the round complexity at all* (and still without relying on random oracles). In contrast, the work of [13] shows a method (used also by [22]) for obtaining universal composability that requires additional messages/rounds. Abdalla et al. [1] show a universally composable PAK protocol, proven secure in the random oracle model, that requires three rounds. To the best of our knowledge, no prior universally composable protocol (whether in the random oracle model or not) can be run in only one round.

1.2 Our Techniques

At a basic level, we rely on smooth projective hash functions [14], as used in [18] (and implicitly in [28]); see Section 2.2 for a definition. The basic structure of previous protocols [28, 18], omitting many important details, is as follows:

First round: The client sends an encryption C of the password pw.

Second round: The server sends an encryption C' of pw, and a projected key $s' = \alpha(k', C, pw)$.

Third round: The client sends a projected key $s = \alpha(k, C', pw)$.

The client computes the session key as $H_k(C', pw) \cdot H_{s'}(C, pw, r)$, and the server computes the session key as $H_s(C', pw, r') \cdot H_{k'}(C, pw)$. (Here, r, r' is the randomness used to compute C, C', respectively.) Properties of the smooth projective hash function ensure that these are equal.

Two difficulties must be overcome in order to collapse a protocol of the above form to one round:

– In the smooth projective hash functions used in prior work, the "projection function" α was *adaptive*, and depended on both the hash key k and the element being hashed (i.e., (C, pw) in the above example). This leads to protocols requiring three rounds just to ensure correctness.

  Here we show a construction of CCA-secure encryption schemes with associated smooth projective hash functions whose projection function is *non-adaptive*, and depends only on the hash key k. This allows us to obtain the *functionality* of PAK in a single round, by having the client send $(\alpha(k), C)$ and the server send $(\alpha(k'), C')$ simultaneously.

– The above addresses correctness, but says nothing about security. The technical difficulty here is that an honestly generated client message $\mathsf{msg} = (s, C)$ might be forwarded by an adversary to *multiple* server instances (and vice versa), and it is required that the session keys computed in all these instances look random and independent to the adversary. (This issue does not arise in prior work because, roughly speaking, messages are *bound* to a single session by virtue of a signature verification key sent in the first round [28, 18] or a MAC derived from the shared session key [17]. Neither approach is viable if we want the entire protocol to take place in a single round.)

  Due to the above difficulty, the proof of security is the most technically challenging part of our work. Our proof relies on a technical lemma related to re-using both the hash keys and the inputs to the smooth projective hash function, and may be of independent interest.

Additional ideas are needed to obtain a *universally composable* protocol without increasing the number of rounds. We refer the reader to Section 5.1 for an overview of the techniques used there.

1.3 Outline of the Paper

In Section 2 we present a standard definition of security for PAK due to Bellare et al. [3]. We also review there the notion of smooth projective hashing, and prove a technical lemma regarding its usage. In Section 3 we describe our basic framework for constructing one-round PAK protocols, and prove security of this approach according to the definition of [3]. We discuss in Section 4 two instantiations of our framework: one based on the decisional Diffie-Hellman assumption, and a second, more efficient instantiation based on bilinear maps. In Section 5 we describe an extension of our framework that yields one-round, universally composable password-based authenticated key-exchange protocols.

2 Definitions and Background

Throughout, we denote the security parameter by n.

2.1 Password-Based Authenticated Key Exchange

Here we present a definition of security for PAK due to Bellare, Pointcheval, and Rogaway [3], based on prior work of [4, 5]. The text here is taken almost verbatim from [28].

Participants, passwords, and initialization. Prior to any execution of the protocol there is an initialization phase during which public parameters and a CRS are established. We assume a fixed set User of protocol participants (also called principals or users). For every distinct $U, U' \in \mathsf{User}$, users U and U' share a password $pw_{U,U'}$. We assume that each $pw_{U,U'}$ is chosen independently and uniformly from the set $[D] \overset{\text{def}}{=} \{1, \dots, D\}$ for some integer D. (Our proof of security extends to more general cases, and we implicitly consider arbitrary password distributions in the setting of universal composability.)

Execution of the protocol. In the real world, a protocol determines how principals behave in response to input from their environment. In the formal model, these inputs are provided by the adversary. Each principal can execute the protocol multiple times (possibly concurrently) with different partners; this is modeled by allowing each principal to have an unlimited number of *instances* with which to execute the protocol. We denote instance i of user U as Π_U^i. Each instance may be used only once. The adversary is given oracle access to these different instances; furthermore, each instance maintains (local) state which is updated during the course of the experiment. In particular, each instance Π_U^i is associated with the following variables:

– sid_U^i, pid_U^i, and sk_U^i denote the *session id*, *partner id*, and *session key* for an instance, respectively. The session id is simply a way to keep track of different executions; we let sid_U^i be the (ordered) concatenation of all messages sent and received by Π_U^i. The partner id denotes the user with whom Π_U^i believes it is interacting. (Note that pid_U^i can never equal U.)
– 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* that 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, updating state as appropriate. The message output by Π_U^i is given to the adversary.
  The adversary can "prompt" instance Π_U^i to initiate the protocol with partner U' by querying $\mathsf{Send}(U, i, U')$. 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 due to, e.g., improper erasure of session keys after use, compromise of a host computer, or cryptanalysis.
- $\mathsf{Test}(U, i)$ — This oracle does not model any real-world capability of the adversary, but is instead used to define security. 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. To be viable, a key-exchange protocol must satisfy the following notion of correctness: if Π_U^i and $\Pi_{U'}^j$ are partnered then $\mathsf{acc}_U^i = \mathsf{acc}_{U'}^j = \text{TRUE}$ and $\mathsf{sk}_U^i = \mathsf{sk}_{U'}^j$, i.e., they both accept and conclude with the same session key.

Advantage of the adversary. Informally, the adversary succeeds if it can guess the bit b used by the Test oracle. To formally define the adversary's success, we first define a notion of *freshness*. 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 $\mathsf{Reveal}(U, i)$; or (2) at some point, the adversary queried $\mathsf{Reveal}(U', j)$, where $\Pi_{U'}^j$ and Π_U^i are partnered. We allow the adversary to succeed only if its Test query is made to a fresh instance; this is necessary for any reasonable definition of security.

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 $b' = b$ (recall that b is the bit chosen by the Test oracle). We denote this event by Succ. The *advantage* of $\mathcal{A}$ in attacking protocol Π is given by $\mathsf{Adv}_{\mathcal{A},\Pi}(k) \stackrel{\text{def}}{=} 2 \cdot \Pr[\mathsf{Succ}] - 1$, where the probability is taken over the random coins used by the adversary and the random coins used during the course of the experiment (including the initialization phase).

It remains to define a secure protocol. A probabilistic polynomial-time (PPT) adversary can always succeed with probability 1 by trying all passwords one-by-one; this is possible since the size of the password dictionary is small. Informally, a protocol is secure if this is the best an adversary can do. Formally, an instance Π_U^i represents an *on-line attack* if both the following are true at the time of the Test query: (1) at some point, the adversary queried $\mathsf{Send}(U, i, *)$; and (2) at some point, the adversary queried $\mathsf{Reveal}(U, i)$ or $\mathsf{Test}(U, i)$. The number of on-line attacks represents a bound on the number of passwords the adversary could have tested in an on-line fashion.

Definition 1. *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* $\mathsf{Adv}_{\mathcal{A},\Pi}(n) \leq Q(n)/D + \mathsf{negl}(n)$.

2.2 Smooth Projective Hash Functions

We provide a self-contained definitional treatment of smooth projective hash functions. These were introduced by Cramer and Shoup [14], and our discussion here is based on that of Gennaro and Lindell [18]. Rather than aiming for utmost generality, we tailor the definitions to our application.

Hard subset membership problems. Fix some integer D. Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a CCA-secure labeled encryption scheme. For a given public key pk, we let C_{pk} denote the set of pairs of valid labels and valid ciphertexts with respect to pk, and require that this set be efficiently recognizable. For a given public key pk, define sets X and $\{L_{pw}\}_{pw \in [D]}$ as follows:

1. $X \overset{\text{def}}{=} \{(\mathsf{label}, C, pw)\}$, where $(\mathsf{label}, C) \in C_{pk}$ and $pw \in \{1, \ldots, D\}$.
2. $L_{pw} \overset{\text{def}}{=} \{(\mathsf{label}, \mathsf{Enc}_{pk}(\mathsf{label}, pw), pw)\}$, where $\mathsf{label} \in \{0,1\}^*$.

Let $L = \bigcup_{pw=1}^{D} L_i$, and note that $L \subset X$. It follows from CCA security of $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ that the following is negligible for any polynomial-time $\mathcal{A}$:

$$\left| \Pr\left[\begin{array}{c} (pk, sk) \leftarrow \mathsf{Gen}(1^n); \\ (\mathsf{label}, pw) \leftarrow \mathcal{A}^{\mathsf{Dec}_{sk}(\cdot,\cdot)}(pk); \\ C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw) \end{array} : \mathcal{A}^{\mathsf{Dec}_{sk}(\cdot,\cdot)}(C) = 1 \right] - \Pr\left[\begin{array}{c} (pk, sk) \leftarrow \mathsf{Gen}(1^n); \\ (\mathsf{label}, pw) \leftarrow \mathcal{A}^{\mathsf{Dec}_{sk}(\cdot,\cdot)}(pk); \\ C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, 0) \end{array} : \mathcal{A}^{\mathsf{Dec}_{sk}(\cdot,\cdot)}(C) = 1 \right] \right|, \qquad (1)$$

where $\mathcal{A}$ is disallowed from querying (label, C) to its decryption oracle.

Smooth projective hash functions. Fix pk and sets $X, \{L_i\}$ as above. A *smooth projective hash function* $\mathcal{H} = \{H_k\}_{k \in K}$ is a keyed function mapping elements in X to elements in some group G, along with a *projection function* $\alpha : K \to S$. Informally, if $x \in L$ then the value of $H_k(x)$ is uniquely determined by $s = \alpha(k)$ and x, whereas if $x \in X \setminus L$ then the value of $H_k(x)$ is statistically close to uniform given $\alpha(k)$ and x (assuming k was chosen uniformly in K). A smooth projective hash function is formally defined by a sampling algorithm that, given pk, outputs $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ such that:

1. There are efficient algorithms for (1) sampling a uniform $k \in K$, (2) computing $H_k(x)$ for $k \in K$ and $x \in X$, and (3) computing $\alpha(k)$ for $k \in K$.
2. For all $(\mathsf{label}, C, pw) \in L$, the value of $H_k(\mathsf{label}, C, pw)$ is uniquely determined by $\alpha(k)$. Moreover, there is an efficient algorithm that takes as input $s = \alpha(k)$ and $(\mathsf{label}, C, pw, r)$ for which $C = \mathsf{Enc}_{pk}(\mathsf{label}, pw; r)$, and outputs $H_k(\mathsf{label}, C, pw)$. (In other words, when $(\mathsf{label}, C, pw) \in L$ then $H_k(\mathsf{label}, C, pw)$ can be computed in two ways: either using k itself, or using $\alpha(k)$ and the randomness used to generate C.)
3. For any (even unbounded) function $f : S \to X \setminus L$, the following distributions have statistical difference negligible in n:

$$\big\{k \leftarrow K; s := \alpha(k) : \big(s, H_k(f(s))\big)\big\} \text{ and } \{k \leftarrow K; s := \alpha(k); g \leftarrow G : (s, g)\}$$

We stress that in the above we modify the definition from [18] in two ways: first, α is *non-adaptive*, and depends on k only (rather than both k and x); second, we require the above to hold even for *adaptive* choice of $f(s) \notin L$. Intuitively, the first modification helps us compress the number of rounds to one, whereas the second is necessary for proving security.

A technical lemma. We now prove a technical lemma regarding smooth projective hash functions. Somewhat informally, Gennaro and Lindell [18] showed that, for randomly generated pk and any label, pw, the distribution

$$\left\{k \leftarrow K; s := \alpha(k); C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw) : \left(s, C, H_k(\mathsf{label}, C, pw)\right)\right\}$$

is computationally indistinguishable from the distribution

$$\{k \leftarrow K; s := \alpha(k); C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw); g \leftarrow G : (s, C, g)\}.$$

(Note this holds even though $H_k(\mathsf{label}, C, pw)$ is uniquely determined by s and C) Here we show that this continues to hold even if hash keys and ciphertexts are *re-used* multiple times. That is, at a high level (ignoring labels and technical details), we show that the distribution

$$\left\{\begin{array}{c} k_1, \ldots, k_\ell \leftarrow K; \forall i : s_i := \alpha(k_i); \\ C_1, \ldots, C_\ell \leftarrow \mathsf{Enc}_{pk}(pw) \end{array} : \left(\{s_i\}, \{C_i\}, \{H_{k_i}(C_j, pw)\}_{i,j=1}^{\ell}\right)\right\}$$

is computationally indistinguishable from the distribution

$$\left\{\begin{array}{c} k_1, \ldots, k_\ell \leftarrow K; \forall i : s_i := \alpha(k_i); \\ C_1, \ldots, C_\ell \leftarrow \mathsf{Enc}_{pk}(pw); g_{i,j} \leftarrow G \end{array} : \left(\{s_i\}, \{C_i\}, \{g_{i,j}\}_{i,j=1}^{\ell}\right)\right\}.$$

Formally, fix a function $\ell = \ell(n)$, let $\mathcal{A}$ be an adversary, and let $b \in \{0, 1\}$. Consider the following experiment Expt_b:

1. Compute $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$ and let $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ be a smooth projective hash function for pk. Give pk to $\mathcal{A}$.
2. Sample $k_1, \ldots, k_\ell \leftarrow K$, and let $s_i := \alpha(k_i)$ for all i. Give $s_1, \ldots, s_\ell$ to $\mathcal{A}$.
3. $\mathcal{A}$ may adaptively query a (modified) encryption oracle that takes as input (label, pw) and outputs a ciphertext $C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw)$ along with
   (a) If $b = 0$, the values $H_{k_i}(\mathsf{label}, C, pw)$ for $i = 1$ to ℓ.
   (b) If $b = 1$, random values $g_1, \ldots, g_\ell \leftarrow G$.
4. $\mathcal{A}$ can also query a decryption oracle $\mathsf{Dec}_{sk}(\cdot, \cdot)$ at any point, except that it may not query any pair (label, C) where C was obtained from the encryption oracle on query (label, pw).
5. At the end of the experiment, $\mathcal{A}$ outputs a bit b'. We say $\mathcal{A}$ *succeeds* if $b' = b$.

A proof of the following appears in the full version of this work [29]:

Lemma 1. *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a CCA-secure labeled encryption scheme. For any polynomial* ℓ *and polynomial-time* $\mathcal{A}$*, we have* $\Pr[\mathcal{A}\ \textit{succeeds}] \leq \frac{1}{2} + \mathsf{negl}(n)$.

$$\begin{array}{rcl}
 & \text{Public parameters: } pk & \\
\underline{\text{User } U} & & \underline{\text{User } U'} \\
k \leftarrow K;\ s := \alpha(k) & & k' \leftarrow K;\ s' := \alpha(k') \\
\mathsf{label} := (U, U', s) & & \mathsf{label}' := (U', U, s') \\
C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw) & \xrightarrow{\quad s, C \quad} & C' \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}', pw) \\
 & \xleftarrow{\quad s', C' \quad} & \\
\mathsf{label}' := (U', U, s') & & \mathsf{label} := (U, U', s) \\
sk_U := H_k(\mathsf{label}', C', pw) & & sk_{U'} := H_k(\mathsf{label}', C', pw) \\
\cdot H_{k'}(\mathsf{label}, C, pw) & & \cdot H_{k'}(\mathsf{label}, C, pw)
\end{array}$$

Fig. 1. A one-round protocol for password-based authenticated key exchange

3 A Framework for One-Round PAK Protocols

Our protocol uses a chosen ciphertext-secure (CCA-secure) labeled public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, and a smooth projective hash function as described in Section 2.2.

Public parameters. The public parameters consist of a public key pk generated by $\mathsf{Gen}(1^n)$. No one need know or store the associated secret key. (For the specific instantiations given in Section 4, a public key can be derived from a common random string.) Let $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ be a smooth projective hash function for pk.

Protocol execution. Consider an execution of the protocol between users U and $U' \neq U$ holding a shared password pw. Our protocol is symmetric, and so we describe the execution from the point of view of U; see also Figure 1.

First, U chooses random hash key $k \leftarrow K$ and computes $s := \alpha(k)$. It then sets $\mathsf{label} := (U, U', s)$ and computes the ciphertext $C \leftarrow \mathsf{Enc}_{pk}(\mathsf{label}, pw)$. It sends the message (s, C).

Upon receiving the message (s', C'), user U does the following. If C' is not a valid ciphertext or $s' \notin S$, then U simply rejects. Otherwise, U sets $\mathsf{label}' := (U', U, s')$ and computes

$$\mathsf{sk}_U := H_k(\mathsf{label}', C', pw) \cdot H_{k'}(\mathsf{label}, C, pw).$$

U computes $H_k(\mathsf{label}', C', pw)$ using k, and can compute $H_{k'}(\mathsf{label}, C, pw)$ using $s' = \alpha(k')$ and the randomness it used to generate C. Correctness follows immediately from the definition of smooth projective hashing.

Theorem 1. *If* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA-secure labeled encryption scheme and* $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ *is a smooth projective hash* *function, then the protocol in Figure 1 is a secure protocol for password-based authenticated key exchange.*

The proof is in the full version of this work [29].

4 Instantiating the Building Blocks

We now discuss two possible instantiations of the building blocks required by the protocol of the previous section. Our first instantiation is based on the decisional Diffie-Hellman (DDH) assumption and (generic) simulation-sound non-interactive zero-knowledge (NIZK) proofs. (It could be based on the quadratic residuosity assumption or the Paillier assumption as well, much like in [18]. We omit further details.) Our second, more efficient construction is based on the decisional linear assumption [8] in groups with a bilinear map.

4.1 A Construction Based on the DDH Assumption

We first describe an encryption scheme and then the associated smooth projective hash function.

A CCA-secure encryption scheme. We construct a CCA-secure encryption scheme by applying the Naor-Yung/Sahai paradigm [32, 35] to the El Gamal encryption scheme. Briefly, the public key defines a group $\mathbb{G}$ of prime order p along with generators $g_1, h_1, g_2, h_2 \in \mathbb{G}$. The public key also contains a common random string crs for a (one-time) simulation-sound NIZK proof system [35].

Fixing $\mathbb{G}$, let $\mathsf{ElGamal}_{g,h}(m)$ denote an El Gamal encryption of $m \in \mathbb{G}$ with respect to (g, h); namely, $\mathsf{ElGamal}_{g,h}(m) \to (g^r,\ h^r \cdot m)$, where $r \in \mathbb{Z}_p$ is chosen uniformly at random. To encrypt a message $m \in \mathbb{G}$ in our CCA-secure scheme, the sender outputs the ciphertext $(\mathsf{ElGamal}_{g_1,h_1}(m),\ \mathsf{ElGamal}_{g_2,h_2}(m),\ \pi)$, where π is a simulation-sound NIZK proof that the same m is encrypted in both cases. Labels can be incorporated by including the label in the proof π; we omit the standard details.

Decryption of the ciphertext $(c_1, d_1, c_2, d_2, \pi)$ rejects if $c_1, d_1, c_2, d_2 \notin \mathbb{G}$ or if the proof π is invalid. (Note that the space of valid label/ciphertext pairs is efficiently recognizable without the secret key.) If the ciphertext is valid, then one of the two component ciphertexts is decrypted and the resulting message is output. The results of [35] show that this yields a CCA-secure (labeled) encryption scheme based on the DDH assumption and simulation-sound NIZK.

A smooth projective hash function. Fix a group $\mathbb{G}$ and a public key $pk = (g_1, h_1, g_2, h_2, \mathsf{crs})$ as above, and define sets X and $\{L_i\}$ as in Section 2.2. Define a smooth projective hash function as follows. The set of keys K consists of all four-tuples of elements in $\mathbb{Z}_p$. Given a valid label/ciphertext pair $(\mathsf{label}, C = (c_1, d_1, c_2, d_2, \pi))$ and key $k = (x_1, y_1, x_2, y_2)$, the hash function is defined as:

$$H_{(x_1,y_1,x_2,y_2)}\big(\mathsf{label}, (c_1, d_1, c_2, d_2, \pi), pw\big) = c_1^{x_1} \cdot (d_1/pw)^{y_1} \cdot c_2^{x_2} \cdot (d_2/pw)^{y_2} .$$

(Thus, the range of H is the group $\mathbb{G}$.) The projection function α is defined as:

$$\alpha(x_1, y_1, x_2, y_2) = (g_1^{x_1} \cdot h_1^{y_1},\ g_2^{x_2} \cdot h_2^{y_2}) .$$

A proof of the following is given in the full version of this work [29].

Lemma 2. *$(K, \mathbb{G}, \mathcal{H} = \{H_k\}_{k \in K}, S, \alpha)$ as defined above is a smooth projective hash function for the hard subset membership problem $(X, \{L_i\})$.*

4.2 A Construction Based on the Decisional Linear Assumption

We now present a more efficient construction based on bilinear maps. The efficiency advantage is obtained by using a *specific* simulation-sound NIZK proof system, built using techniques adapted from [23,11]. Our construction here relies on the *decisional linear assumption* as introduced by Boneh et al. [8]; we refer the reader there for a precise statement of the assumption.

A CPA-secure encryption scheme. We start by describing a semantically secure encryption scheme, due to Boneh et al. [8], based on the decisional linear assumption; we then convert this into a CCA-secure encryption scheme via the same paradigm as above, but using an efficient simulation-sound NIZK proof system. The bilinear map itself is used only in the construction of the simulation-sound NIZK.

Fix groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The public key is $pk = (f, g, h) \in \mathbb{G}^3$, and the secret key is (α, β) such that $f = h^{1/\alpha}$ and $g = h^{1/\beta}$. A message $m \in \mathbb{G}$ is encrypted by choosing random $r, s \in \mathbb{Z}_p$ and computing the ciphertext $(f^r, g^s, h^{r+s} \cdot m)$. Given a ciphertext (c_1, c_2, c_3), we can recover m as $c_3/c_1^{\alpha}c_2^{\beta}$.

A simulation-sound NIZK proof of plaintext equality. We can construct a (one-time) simulation-sound NIZK proof of plaintext equality for the encryption scheme described above using the techniques of [23,11]. Details of the construction (which, while not entirely straightforward, are not the focus of this work) are given in Appendix A.

A CCA-secure encryption scheme. We obtain a CCA-secure encryption scheme by using the Naor-Yung/Sahai paradigm, as described previously. (The following discussion relies on the results of Appendix A.) The public key consists of group elements (f_1, g_1, f_2, g_2, h) used for encryption, in addition to any group elements needed for the CRS of the simulation-sound NIZK proof. Encryption of m, as described in Appendix A, is done by choosing $r_1, s_1, r_2, s_2 \in \mathbb{Z}_p$ and computing the ciphertext

$$(f_1^{r_1},\ g_1^{s_1},\ h^{r_1+s_1} \cdot m,\ f_2^{r_2},\ g_2^{s_2},\ h^{r_2+s_2} \cdot m,\ \pi),$$

where π denotes a simulation-sound NIZK proof that the same m was encrypted both times. (Once again, the space of valid label/ciphertext pairs is efficiently recognizable without the secret key.) It follows from [32, 35] that this yields a

CCA-secure scheme under the decisional linear assumption. Ciphertexts consist of 66 group elements altogether.

A smooth projective hash function. Fix $\mathbb{G}, \mathbb{G}_T$, and a public-key $pk = (f_1, g_1, f_2, g_2, h)$ as above, and define sets X and $\{L_i\}$ as in Section 2.2. We define a smooth projective hash function as follows. The set of keys K is the set of six-tuples of elements in $\mathbb{Z}_p$. Given a valid label/ciphertext pair $(\mathsf{label}, C = (c_1, d_1, e_1, c_2, d_2, e_2, \pi))$ and a key $k = (x_1, y_1, z_1, x_2, y_2, z_2) \in \mathbb{Z}_p^6$, the hash function is defined as

$$H_{(x_1,y_1,z_1,x_2,y_2,z_2)}(\mathsf{label}, C, pw) = c_1^{x_1} \cdot d_1^{y_1} \cdot (e_1/pw)^{z_1} \cdot c_2^{x_2} \cdot d_2^{y_2} \cdot (e_2/pw)^{z_2} .$$

(The range of H is $\mathbb{G}$ itself.) The projection function $\alpha : K \to \mathbb{G}^4$ is defined as:

$$\alpha(x_1, y_1, z_1, x_2, y_2, z_2) = (f_1^{x_1} h^{z_1},\ g_1^{y_1} h^{z_1},\ f_2^{x_2} h^{z_2},\ g_2^{y_2} h^{z_2}) .$$

In Appendix B we show:

Lemma 3. *$(K, \mathbb{G}, \mathcal{H} = \{H_k\}_{k \in K}, S, \alpha)$ as defined above is a smooth projective hash function for the hard subset membership problem $(X, \{L_i\})$.*

5 A One-Round, Universally-Composable PAK Protocol

Canetti et al. [13] gave a definition of security for password-based authenticated key exchange in the universal composability (UC) framework [12]. Their definition guarantees a strong, simulation-based notion of security that, in particular, guarantees that security is maintained even when multiple protocols are run concurrently in an arbitrary network. For the specific case of password-based key exchange, the definition also has the advantage of *automatically* handling arbitrary (efficiently sampleable) distributions on passwords, and even correlations between passwords of different users. We refer to [13] for a more complete discussion, and a description of the password-based key-exchange functionality $\mathcal{F}_{\mathsf{pwKE}}$. We let $\hat{\mathcal{F}}_{\mathsf{pwKE}}$ denote the multi-session extension of $\mathcal{F}_{\mathsf{pwKE}}$.

5.1 Overview of the Construction

We do not know how to prove that the protocol from Section 3 is universally composable. The main difficulty is that the definition of PAK in the UC framework requires simulation *even if the adversary guesses the correct password.* (In contrast, in the proof of security in Section 3 we simply "give up" in case this ever occurs.) To see the problem more clearly, consider what happens in the UC setting when the simulator sends the first message of the protocol to the adversary, before the simulator knows the correct password. The simulator must send *some* ciphertext C as part of the first message, and this "commits" the simulator to some password pw. When the adversary sends the reply, the simulator can extract the adversary's "password guess" pw' and submit this guess to the ideal functionality. If this turns out to be the correct password, however, the simulator is stuck: it needs to compute a session key that matches the session key the adversary would compute, but the simulator is (information-theoretically!) unable to do so because it sent an incorrect ciphertext in the first message.

In prior work [13], the issue above was resolved by having one party send a "pre-commitment" to the password, and then running a regular PAK protocol along with a proof that the password being used in the protocol is the same as the password to which it "pre-committed". (The proof is set up in such a way that the simulator can equivocate this proof, but the adversary cannot.) This requires at least one additional round.

We take a different approach that does not affect the round complexity at all. Roughly, we modify the protocol from Figure 1 by having each party include as part of its message an encryption C_1 of its hash key k, along with a proof that C_1 encrypts a value k for which $\alpha(k) = s$. Now, even if the simulator is wrong in its guess of the password it will still be able to compute a session key by extracting this hash key from the adversary's message. A full description of the protocol is given in the following section.

While we do not describe in detail any instantiation of the components, we remark that it should be possible to use the same techniques as in Appendix A to construct (reasonably) efficient realizations of the necessary components using bilinear maps. We leave this for future work.

5.2 Description of the Protocol

In addition to the building blocks used in Section 3, here we also rely on an unbounded simulation-sound [15] NIZK proof system $(\mathsf{CRSGen}, \mathcal{P}, \mathcal{V})$ for a language L^* defined below.

Public parameters. The public parameters consist of two public keys pk_1, pk_2 generated by $\mathsf{Gen}(1^n)$ and a common random string crs for the simulation-sound NIZK proof system. Let $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ be a smooth projective hash function for pk_2.

Protocol execution. Consider an execution of the protocol between users U and $U' \neq U$ holding a shared password pw and a common session identifier ssid. (The ssid is an artifact of the UC framework, and it is guaranteed that (1) parties communicating with each other begin holding matching ssids, and (2) each ssid is used only once. Existence of these ssids is not essential to our proof of security, though it does make the proof somewhat simpler.) Our protocol is symmetric, and so we describe the execution from the point of view of U; see Figure 2.

First, U chooses a random hash key $k \leftarrow K$ and computes $s := \alpha(k)$. It then computes an encryption of k, namely $C_1 \leftarrow \mathsf{Enc}_{pk_1}(k)$. Define a language L^* as follows.

$$L^* \overset{\text{def}}{=} \{(s, C_1) : \exists k \in K \text{ and } \omega \text{ s.t } s = \alpha(k) \text{ and } C_1 = \mathsf{Enc}_{pk_1}(k; \omega)\}.$$

U computes an NIZK proof π that $(C_1, s) \in L^*$, using crs. It then sets $\mathsf{label} := (\mathsf{ssid}, U, U', s, C_1, \pi)$ and computes the ciphertext $C_2 \leftarrow \mathsf{Enc}_{pk_2}(\mathsf{label}, pw)$. The message it sends is (s, C_1, π, C_2).

Public Parameters: $(pk_1, pk_2, \mathsf{crs})$

User U		User U'
$k \leftarrow K;\ s := \alpha(k)$		$k' \leftarrow K;\ s' := \alpha(k')$
$C_1 \leftarrow \mathsf{Enc}_{pk_1}(k)$		$C_1' \leftarrow \mathsf{Enc}_{pk_1}(k')$
$\pi := \mathcal{P}_{\mathsf{crs}}((s, C_1) \in L^*)$		$\pi' := \mathcal{P}_{\mathsf{crs}}((s', C_1') \in L^*)$
$\mathsf{label} := (\mathsf{ssid}, U, U', s, C_1, \pi)$		$\mathsf{label}' := (\mathsf{ssid}, U', U, s', C_1', \pi')$
$C_2 \leftarrow \mathsf{Enc}_{pk_2}(\mathsf{label}, pw)$	$\xrightarrow{s, C_1, \pi, C_2}$	$C_2' \leftarrow \mathsf{Enc}_{pk_2}(\mathsf{label}', pw)$
	$\xleftarrow{s', C_1', \pi', C_2'}$	
$\mathsf{label}' := (\mathsf{ssid}, U', U, s', C_1', \pi')$		$\mathsf{label} := (\mathsf{ssid}, U, U', s, C_1, \pi)$
$sk_U := H_k(\mathsf{label}', C_2', pw)$		$sk_{U'} := H_k(\mathsf{label}', C_2', pw)$
$\cdot H_{k'}(\mathsf{label}, C_2, pw)$		$\cdot H_{k'}(\mathsf{label}, C_2, pw)$

Fig. 2. A universally composable protocol for password-based authenticated key exchange

Upon receiving the message (s', C_1', π', C_2'), user U does the following. If the message is invalid (i.e., if verification of π' fails, or C_2' is not a valid ciphertext, or $s' \notin S$), then U simply rejects. Otherwise, U sets $\mathsf{label}' := (\mathsf{ssid}, U', U, s', C_1', \pi')$ and computes $\mathsf{sk}_U := H_k(\mathsf{label}', C_2', pw) \cdot H_{k'}(\mathsf{label}, C_2, pw)$. Note U can compute $H_k(\mathsf{label}', C_2', pw)$ since it knows k, and can compute $H_{k'}(\mathsf{label}, C_2, pw)$ using $s' = \alpha(k')$ and the randomness used to generate C_2. Correctness follows from the definition of smooth projective hashing. A proof of the following is given the full version of this work [29].

Theorem 2. *If* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA-secure public-key encryption scheme,* $(\mathsf{CRSGen}, \mathcal{P}, \mathcal{V})$ *is an unbounded simulation-sound NIZK proof system, and furthermore* $(K, G, \mathcal{H} = \{H_k : X \to G\}_{k \in K}, S, \alpha : K \to S)$ *is a smooth projective hash family, then the protocol in Figure 2 securely realizes* $\hat{\mathcal{F}}_{\mathsf{pwKE}}$ *in the* $\mathcal{F}_{\mathsf{crs}}$*-hybrid model.*

References

1. Abdalla, M., Catalano, D., Chevalier, C., Pointcheval, D.: Efficient two-party password-based key exchange protocols in the UC framework. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 335–351. Springer, Heidelberg (2008)
2. Barak, B., Canetti, R., Lindell, Y., Pass, R., Rabin, T.: Secure computation without authentication. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 361–377. Springer, Heidelberg (2005)

3. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
4. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
5. Bellare, M., Rogaway, P.: Provably secure session key distribution: The three party case. In: 27th Annual ACM Symposium on Theory of Computing (STOC), pp. 57–66. ACM Press, New York (1995)
6. Bellovin, S.M., Merritt, M.: Encrypted key exchange: Password-based protocols secure against dictionary attacks. In: IEEE Symposium on Security & Privacy, pp. 72–84. IEEE, Los Alamitos (1992)
7. Bird, R., Gopal, I., Herzberg, A., Janson, P., Kutten, S., Molva, R., Yung, M.: Systematic design of two-party authentication protocols. IEEE J. on Selected Areas in Communications 11(5), 679–693 (1993)
8. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
9. Boyen, X., Dodis, Y., Katz, J., Ostrovsky, R., Smith, A.: Secure remote authentication using biometric data. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 147–163. Springer, Heidelberg (2005)
10. Boyko, V., MacKenzie, P.D., Patel, S.: Provably secure password-authenticated key exchange using diffie-hellman. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 156–171. Springer, Heidelberg (2000)
11. Camenisch, J., Chandran, N., Shoup, V.: A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 351–368. Springer, Heidelberg (2009), `http://eprint.iacr.org/2008/375`
12. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd Annual Symposium on Foundations of Computer Science (FOCS), pp. 136–145. IEEE, Los Alamitos (2001)
13. Canetti, R., Halevi, S., Katz, J., Lindell, Y., MacKenzie, P.D.: Universally composable password-based key exchange. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 404–421. Springer, Heidelberg (2005)
14. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
15. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001)
16. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Information Theory 22(6), 644–654 (1976)
17. Gennaro, R.: Faster and shorter password-authenticated key exchange. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 589–606. Springer, Heidelberg (2008)
18. Gennaro, R., Lindell, Y.: A framework for password-based authenticated key exchange. ACM Trans. Information and System Security 9(2), 181–234 (2006)
19. Goldreich, O., Lindell, Y.: Session-key generation using human passwords only. Journal of Cryptology 19(3), 241–340 (2006)
20. Gong, L., Lomas, T.M.A., Needham, R.M., Saltzer, J.H.: Protecting poorly chosen secrets from guessing attacks. IEEE J. Selected Areas in Communications 11(5), 648–656 (1993)

21. Goyal, V., Jain, A., Ostrovsky, R.: Password-authenticated session-key generation on the internet in the plain model. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 277–294. Springer, Heidelberg (2010)
22. Groce, A., Katz, J.: A new framework for efficient password-based authenticated key exchange. In: 17th ACM Conf. on Computer and Communications Security (CCCS), pp. 516–525. ACM Press, New York (2010)
23. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
24. Halevi, S., Krawczyk, H.: Public-key cryptography and password protocols. ACM Trans. Information and System Security 2(3), 230–268 (1999)
25. Jeong, I.R., Katz, J., Lee, D.-H.: One-round protocols for two-party authenticated key exchange. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 220–232. Springer, Heidelberg (2004)
26. Jiang, S., Gong, G.: Password based key exchange with mutual authentication. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 267–279. Springer, Heidelberg (2004)
27. Katz, J., MacKenzie, P.D., Taban, G., Gligor, V.D.: Two-server password-only authenticated key exchange. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 1–16. Springer, Heidelberg (2005)
28. Katz, J., Ostrovsky, R., Yung, M.: Efficient password-authenticated key exchange using human-memorable passwords. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 475–494. Springer, Heidelberg (2001)
29. Katz, J., Vaikuntanathan, V.: Round-optimal password-based authenticated key exchange, `http://eprint.iacr.org/2010/368`
30. Katz, J., Vaikuntanathan, V.: Smooth projective hashing and password-based authenticated key exchange from lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 636–652. Springer, Heidelberg (2009)
31. MacKenzie, P.D., Patel, S., Swaminathan, R.: Password-authenticated key exchange based on RSA. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 599–613. Springer, Heidelberg (2000)
32. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: 21st Annual ACM Symposium on Theory of Computing (STOC), pp. 33–43. ACM Press, New York (1989)
33. Nguyen, M.-H., Vadhan, S.: Simpler session-key generation from short random passwords. Journal of Cryptology 21(1), 52–96 (2008)
34. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
35. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th Annual Symposium on Foundations of Computer Science (FOCS), pp. 543–553. IEEE, Los Alamitos (1999)

A A Simulation-Sound NIZK Proof of Plaintext Equality

Fix groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ as in Section 4.2. Fix also two public keys $pk_1 = (f_1, g_1, h)$ and $pk_2 = (f_2, g_2, h)$. We encrypt a message m with respect to pk_1 by choosing random r, s and computing the ciphertext $(f_1^r, g_1^s, h^{r+s} \cdot m)$. We encrypt a message m with respect to pk_2 by choosing random $r, s \in \mathbb{Z}_p$ and computing the ciphertext $(f_2^r, g_2^s, h^{r+s} \cdot m)$. We stress that the public keys use the same value h.

We first describe a (potentially malleable) NIZK proof of plaintext equality. That is, given two ciphertexts (F_1, G_1, H_1) and (F_2, G_2, H_2) encrypted with respect to pk_1, pk_2, respectively, we describe a proof that these ciphertexts encrypt the same message. The observation is that plaintext equality is equivalent to the existence of $r_1, s_1, r_2, s_2 \in \mathbb{Z}_p$ such that:

$$F_1 = f_1^{r_1} \tag{2}$$
$$G_1 = g_1^{s_1} \tag{3}$$
$$F_2 = f_2^{r_2} \tag{4}$$
$$G_2 = g_2^{s_2} \tag{5}$$
$$H_1/H_2 = h^{r_1+s_1-r_2-s_2}. \tag{6}$$

As shown in [23] (see also [11, Section 4.4] for a self-contained description), NIZK proofs of satisfiability (with a CRS) can be constructed for a system of equations as above; since, in our case, we have 5 (linear) equations in 4 variables, proofs contain 22 group elements[2].

Camenisch et al. [11] show a construction of an *unbounded* simulation-sound NIZK. For our purposes, a simpler construction that is *one-time* simulation sound [35] suffices. Let (Gen, Sign, Vrfy) be a one-time signature scheme, where for simplicity we assume verification keys are elements of $\mathbb{G}$ (this can always be achieved using an extra step of hashing). To make the above (one-time) simulation-sound, we add group elements (f, g, h, F, G, H) to the CRS. Roughly, proofs of plaintext equality now contain:

1. A fresh signature verification key vk.
2. A proof that *either* there exists a satisfying assignment to Equations (2)–(6), *or* that the given tuple (f, g, h, F, G, H) is an encryption of vk. I.e., there exist r, s such that:
$$F = f^r, \qquad G = g^s, \qquad H/vk = h^{r+s}. \tag{7}$$
3. A signature σ (with respect to vk) on the proof from the previous step.

Noting that Equation (7) describes a system of 3 (linear) equations in 2 variables, and using the techniques from [11, Appendix A.2], an NIZK proof as required in step 2 can be done using 58 group elements, for a total of 60 group elements for the entire simulation-sound NIZK proof (assuming signatures are one group element for simplicity). See also footnote 2.

B Proof of Lemma 3

Sampling a uniform $k \in K$, computing $H_k(x)$ given $k \in K$ and $x \in X$ and computing $\alpha(k)$ for $k \in K$ are all easy.

[2] Our calculations here are based on the decisional linear assumption (the 2-linear assumption in the terminology of [11]). If we are willing to use the 1-linear assumption, the efficiency of our proofs can be improved.

We show that if $(\mathsf{label}, C, pw) \in L$, then $H_k(\mathsf{label}, C, pw)$ can be computed efficiently given $\alpha(k)$ and the randomness that was used to generate C. Since $(\mathsf{label}, C, pw) \in L$, we have that $C = (f_1^{r_1},\, g_1^{s_1},\, h^{r_1+s_1} f_1^{pw},\, f_2^{r_2},\, g_2^{s_2},\, h^{r_2+s_2} f_2^{pw})$ for some $r_1, s_1, r_2, s_2 \in \mathbb{Z}_p$. For $k = (x_1, y_1, z_1, x_2, y_2, z_2)$ we have

$$\begin{aligned} H_k(\mathsf{label}, C, pw) &= c_1^{x_1} d_1^{y_1} \cdot (e_1/f_1^{pw})^{z_1} \cdot c_2^{x_2} \cdot d_2^{y_2} \cdot (e_2/f_2^{pw})^{z_2} \\ &= (f_1^{x_1} h^{z_1})^{r_1} \cdot (g_1^{y_1} h^{z_1})^{s_1} \cdot (f_2^{x_2} h^{z_2})^{r_2} \cdot (g_2^{y_2} h^{z_2})^{s_2}\,. \end{aligned}$$

This can be computed easily given r_1, s_1, r_2, s_2, and

$$\alpha(k) \stackrel{\text{def}}{=} (f_1^{x_1} h^{z_1},\; g_1^{y_1} h^{z_1},\; f_2^{x_2} h^{z_2},\; g_2^{y_2} h^{z_2})\,.$$

Next, we show that if $(\mathsf{label}, C, pw) \in X \setminus L$, then the value of $H_k(\mathsf{label}, C, pw)$ is uniform conditioned on $\alpha(k)$. (This holds even if (label, C, pw) are chosen adaptively depending on $\alpha(k)$.) Fix any $\alpha(k) = (S_1, S_2, S_3, S_4)$. Letting $\alpha_i = \log_h f_i$ and $\beta_i = \log_h g_i$, this value of $\alpha(k)$ constrains $k = (x_1, y_1, z_1, x_2, y_2, z_2)$ to satisfy

$$\begin{pmatrix} \alpha_1 & 0 & 1 & 0 & 0 & 0 \\ 0 & \beta_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_2 & 0 & 1 \\ 0 & 0 & 0 & 0 & \beta_2 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \end{pmatrix}, \tag{8}$$

where $\gamma_i = \log_h S_i$. For any $(\mathsf{label}, C, pw) \in X \setminus L$, we can write

$$C = (f_1^{r_1},\, g_1^{s_1},\, h^{r_1+s_1} f_1^{pw'},\, f_2^{r_2},\, g_2^{s_2},\, h^{r_2+s_2} f_2^{pw'},\, \pi)$$

for some $pw' \neq pw$. (We assume for simplicity that the same pw' is encrypted twice; since π is valid, this is the case with all but negligible probability.) We then have

$$\begin{aligned} &H_k(\mathsf{label}, C, pw) \\ &= f_1^{r_1 x_1} \cdot g_1^{s_1 y_1} \cdot h^{(r_1+s_1) z_1} \cdot \left(f_1^{\Delta}\right)^{z_1} \cdot f_2^{r_2 x_2} \cdot g_2^{s_2 y_2} \cdot h^{(r_2+s_2) z_2} \cdot \left(f_2^{\Delta}\right)^{z_2} \\ &= S_1^{r_1} S_2^{s_1} S_3^{r_2} S_4^{s_2} \cdot (f_1^{z_1} f_2^{z_2})^{\Delta}\,, \end{aligned} \tag{9}$$

where $\Delta = pw' - pw \neq 0$. For any $g \in \mathbb{G}$, we have $f_1^{z_1} f_2^{z_2} = g$ iff

$$\alpha_1 \cdot z_1 + \alpha_2 \cdot z_2 = \log_h g. \tag{10}$$

Since the system of equations given by (8) and (10) is under-defined, the probability that $f_1^{z_1} f_2^{z_2} = g$ is exactly $1/|\mathbb{G}|$ even conditioned on the value $\alpha(k)$. Looking at Equation (9), and noting that $S_1^{r_1} S_2^{s_1} S_3^{r_2} S_4^{s_2}$ is determined by $\alpha(k)$ and C, we conclude that the distribution of $H_k(\mathsf{label}, C, pw)$ is uniform in $\mathbb{G}$.