# Simpler Session-Key Generation from Short Random Passwords

Minh-Huyen Nguyen [⋆] and Salil Vadhan[⋆⋆]

Harvard University, Cambridge, MA
{mnguyen,salil}@eecs.harvard.edu

**Abstract.** Goldreich and Lindell (CRYPTO '01) recently presented the first protocol for password-authenticated key exchange in the standard model (with no common reference string or set-up assumptions other than the shared password). However, their protocol uses several heavy tools and has a complicated analysis.

We present a simplification of the Goldreich–Lindell (GL) protocol and analysis for the special case when the dictionary is of the form $\mathcal{D} = \{0,1\}^d$, i.e. the password is a short random string (like an ATM PIN number). Our protocol can be converted into one for arbitrary dictionaries using a common reference string of logarithmic length. The security bound achieved by our protocol is somewhat worse than the GL protocol. Roughly speaking, our protocol guarantees that the adversary can "break" the scheme with probability at most $O(\mathrm{poly}(n)/|\mathcal{D}|)^{\Omega(1)}$, whereas the GL protocol guarantees a bound of $O(1/|\mathcal{D}|)$.

We also present an alternative, more natural definition of security than the "augmented definition" of Goldreich and Lindell, and prove that the two definitions are equivalent.

## 1 Introduction

*What is the minimal amount of information that two parties must share in order to perform nontrivial cryptography?* This fundamental question is at the heart of many of the major distinctions we draw in cryptography. Classical private-key cryptography assumes that the legitimate parties share a long random key. Public-key cryptography mitigates this by allowing the sharing of information to be done through public keys that need not be hidden from the adversary. However, in both cases, the amount of information shared by the legitimate parties (e.g. as measured by mutual information) needs to be quite large. Indeed, the traditional view is that security comes from the adversary's inability to exhaustively search the keyspace.

Thus it is very natural to ask: *can we do nontrivial cryptography using "low-entropy" keys?* That is, using a keyspace that is feasible to exhaustively search.

[⋆] Supported by NSF grant CCR-0205423.

[⋆⋆] Supported by NSF grant CCR-0205423 and a Sloan Research Fellowship. Part of this work done while at the Radcliffe Institute for Advanced Study.

In addition to being a natural theoretical question, it has clear relevance to the many "real-life" situations where we need security but only have a low-entropy key (e.g. an ATM PIN number, or human-chosen password on a website).

Public-key cryptography provides an initial positive answer to this question: key-exchange protocols, as in [10], do not require *any* prior shared information. However, this holds only for passive adversaries, and it is well known that without any prior shared information between the legitimate parties, an active adversary can always succeed through a person-in-the-middle attack. Thus, it remains an interesting question to achieve security against active adversaries using a low-entropy shared key. This has led researchers to consider the problem of *password-authenticated key exchange,* which we describe next.

*Password-Authenticated Key Exchange.* The password-authenticated key exchange problem was first suggested by Bellovin and Merritt [4]. We assume that two parties, Alice and Bob, share a password $w$ chosen uniformly at random from a dictionary $\mathcal{D} \subseteq \{0,1\}^n$. This dictionary can be very small, e.g. $|\mathcal{D}| = \text{poly}(n)$, and in particular it may be feasible for an adversary to exhaustively search it. Our aim is to construct a protocol enabling Alice and Bob to generate a "random" session key $K \in \{0,1\}^n$, which they can subsequently use for standard private-key cryptography. We consider an active adversary that completely controls the communication channel between Alice and Bob. The adversary can intercept, modify, drop, and delay messages, and in particular can attempt to impersonate either party through a person-in-the-middle attack.

Our goal is that, even after the adversary mounts such an attack, Alice and Bob will generate a session key that is indistinguishable from uniform even given the adversary's view. However, our ability to achieve this goal is limited by two unpreventable attacks. First, since the adversary can block all communication, it can prevent one or both of the parties from completing the protocol and obtaining a session key. Second, the adversary can guess a random password $\tilde{w} \leftarrow \mathcal{D}$ and attempt to impersonate one of the parties. With probability $1/|\mathcal{D}|$, the guess equals the real password (i.e., $\tilde{w} = w$), and the adversary will succeed in the impersonation and therefore learn the session key. Thus, we revise our goal to effectively limit the adversary to these two attacks. Various formalizations for this problem have been developed through several works [3,15,22,2,7,12]. We follow the definitional framework of Goldreich and Lindell [12], which is described in more detail in Sec. 2.

In addition to addressing what can be done with a minimal amount of shared information, the study of this problem is useful as another testbed for developing our understanding of *concurrency* in cryptographic protocols. The concurrency implicitly arises from the person-in-the-middle attack, which we can view as two simultaneous executions of the protocol, one between Alice and the adversary and the other between Bob and the adversary.

The first protocols for the password-authenticated key exchange problem were proposed in the security literature, based on informal definitions and heuristic arguments (e.g. [5,24]). The first rigorous proofs of security were given in the random oracle model [2,7]. Only recently were rigorous solutions without random oracles given, in independent works by Goldreich and Lindell [12] and Katz, Ostrovsky, and Yung [16]. One of the main differences between these two protocols is that the KOY protocol (and the subsequent protocols of [17,11]) is in the "public parameters model," requiring a string to be generated and published by a trusted third party, whereas the GL protocol requires no set-up assumption other than the shared password. Thus, even though the KOY protocol has a number of practical and theoretical advantages over the GL protocol (which we will not enumerate here), the GL protocol is more relevant to our initial question about the minimal amount of shared information needed for nontrivial cryptography.

*The Goldreich–Lindell Protocol.* As mentioned above, the Goldreich–Lindell protocol [12] is remarkable in that the only set-up assumption it requires is that the two parties share a password chosen at random from an arbitrary dictionary. Their protocol can be based on general complexity assumptions (the existence of trapdoor permutations), can be implemented in a constant number of rounds (under stronger assumptions), and achieves a nearly optimal security bound (the adversary has probability only $O(1/|\mathcal{D}|)$ of "breaking" the scheme).

Despite giving such a strong result, the Goldreich–Lindell protocol does not leave us with a complete understanding of the password-authenticated key exchange problem. First, the protocol makes use of several "heavy" tools: secure two-party polynomial evaluation (building on [19], who observed that this yields a protocol for password-authenticated key exchange against passive adversaries), nonmalleable commitments (as suggested in [6]), and the specific concurrent zero-knowledge proof of Richardson and Kilian [21]. It is unclear whether all of these tools are really essential for solving the key exchange problem. Second, the proof of the protocol's security is extremely complicated. Goldreich and Lindell do introduce nice techniques for analyzing concurrent executions (arising from the person-in-the-middle attack) of two-party protocols whose security is only guaranteed in the stand-alone setting (e.g. the polynomial evaluation). But these techniques are applied in an intricate manner that seems inextricably tied to the presence of the nonmalleable commitment and zero-knowledge proof. Finally, finding an efficient instantiation of the Goldreich–Lindell protocol would require finding efficient instantiations of all three of the heavy tools mentioned above, which seems difficult. In particular, the Richardson-Kilian zero-knowledge proof is used to prove an **NP** statement that asserts the consistency of a transcript of the nonmalleable commitment, a standard commitment, and the output of an iterated one-way permutation. For such an **NP** statement, it seems difficult to avoid using a generic zero-knowledge proof system for **NP**, which are almost always inefficient due to the use of Cook's theorem.

*Our Protocol.* Our main result is a simplification of the Goldreich–Lindell protocol and analysis for the special case when the dictionary is of the form $\mathcal{D} = \{0,1\}^d$, i.e. the password is a short random string (like an ATM PIN number). This special case still retains many of the key features of the problem: the person-in-the-middle attack and the resulting concurrency issues are still present, and the adversary can still exhaustively search the dictionary (since we allow the password length $d$ to be as small as $O(\log n)$, where $n$ is the security parameter). Moreover, our protocol can be converted into one for arbitrary dictionaries in the common reference string model (using the common reference string as the seed of a randomness extractor [20]). For dictionaries $\mathcal{D} \subset \{0,1\}^n$, the common reference string is a uniform string of only logarithmic length (specifically, $O(\log n + \log |\mathcal{D}|)$), and thus retains the spirit of minimizing the amount of shared information between the legitimate parties. In contrast, the previous protocols in the public parameters model [16,17,11] require a public string of length poly$(n)$ with special number-theoretic structure.

The main way in which we simplify the GL protocol is that we remove the nonmalleable commitments and the Richardson–Kilian zero-knowledge proof. Instead, our protocol combines secure polynomial evaluation with a combinatorial tool (almost pairwise independent hashing), in addition to using "lightweight" cryptographic primitives also used in [12] (one-way permutations, one-time MACs, standard commitments). Our analysis is also similarly simpler. While it has the same overall structure as the analysis in [12] and utilizes their techniques for applying the stand-alone properties of the polynomial evaluation in the concurrent setting, it avoids dealing with the nonmalleable commitments and the zero-knowledge proof (which is the most complex part of the GL analysis).

Removing the nonmalleable commitments and the RK zero-knowledge proof has two additional implications. First, finding an efficient implementation of our protocol only requires finding an efficient protocol for secure polynomial evaluation (in fact, only for linear polynomials).[1] Since this is a highly structured special case of secure two-party computation, it does not seem beyond reach to find an efficient protocol. Indeed, Naor and Pinkas [19] have already given an efficient polynomial evaluation protocol for passive adversaries. Second, our protocol can be implemented in a constant number of rounds assuming only the existence of trapdoor permutations, whereas implementing the Goldreich–Lindell protocol in a constant number of rounds requires additional assumptions, such as the existence of claw-free permutations (for [21]) and some sort of exponential hardness assumption (to use [1]).

We note that the security bound achieved by our protocols is somewhat worse than in previous works. Roughly speaking, our protocol guarantees that the adversary can "break" the scheme with probability at most $O\left(\frac{\text{poly}(n)}{|\mathcal{D}|}\right)^{\Omega(1)}$, whereas previous works guarantee a bound of $O(1/|\mathcal{D}|)$.

An additional result in our paper involves the definition of security in [12]. As pointed out by Rackoff (cf., [2]), it is important that a key exchange protocol provide security even if the party who completes the protocol first starts using the generated key in some application before the second party completes the protocol. In order to address this issue, Goldreich and Lindell [12] augmented

[1] Actually, we require a slightly augmented form of polynomial evaluation, in which one of the parties commits to its input beforehand and the protocol ensures consistency with this committed input.

their definition with a "session-key challenge", in which the adversary is given either the generated key or a uniform string with probability 1/2 upon the first party's completion of the protocol. We present an arguably more natural definition that directly models the use of the generated key in an arbitrary application, and prove its equivalence to the augmented definition of Goldreich and Lindell [12]. (This result is analogous to the result of Shoup [22] for non-password-based key exchange protocols.)

## 2 Definition of Security

We adopt the notation of Goldreich and Lindell and refer the reader to [12] for more details.

- $C$ denotes the probabilistic polynomial time adversary through which the honest parties $A$ and $B$ communicate. We model this communication by giving $C$ oracle access to a single copy of $A$ and a single copy of $B$. Here the oracles $A$ and $B$ have memory and represent honest parties executing the session-key generation protocol. We denote by $C^{A(x),B(y)}(\sigma)$ an execution of $C$ with auxiliary input $\sigma$ when it communicates with $A$ and $B$, with respective inputs $x$ and $y$. The output of the channel $C$ from this execution is denoted by output $\left(C^{A(x),B(y)}(\sigma)\right)$.
- The security parameter is denoted by $n$. The password dictionary is denoted by $\mathcal{D} \subseteq \{0,1\}^n$ and we write $\epsilon = \frac{1}{|\mathcal{D}|}$.

We denote by $U_n$ the uniform distribution over strings of length $n$, by $\text{neg}(n)$ a negligible function and write $x \overset{\text{R}}{\leftarrow} S$ when $x$ is chosen uniformly from the set $S$.

For a function $\gamma : \mathbb{N} \to [0,1]$, we say that the probability ensembles $\{X_n\}$ and $\{Y_n\}$ are $(1-\gamma)$-*indistinguishable* (denoted by $\{X_n\} \overset{\gamma}{\equiv} \{Y_n\}$) if for every nonuniform PPT distinguisher $D$ and all $n$,

$$|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| < \gamma(n) + \text{neg}(n) \ .$$

We say that $\{X_n\}$ and $\{Y_n\}$ are *computationally indistinguishable*, which we denote by $X_n \overset{\text{c}}{\equiv} Y_n$, if they are 1-indistinguishable. We say that $\{X_n\}$ is $(1-\gamma)$ *pseudorandom* if it is $(1-\gamma)$ indistinguishable from $U_n$.

We will now formalize the problem of session-key generation using human passwords. We first follow the presentation of the problem as in [12] and then contrast it with our definition.

### 2.1 The Initial Definition

The definition in [12] follows the standard paradigm for secure computation: define an ideal functionality (using a trusted third party) and require that every adversary attacking the real protocol can be simulated by an ideal adversary attacking the ideal functionality. Note that in the real protocol, the active adversary $C$ can prevent one or both of the parties $A$ and $B$ from having an output. Thus, in the ideal model, we will allow $C_{\text{ideal}}$ to specify two input bits, $\text{dec}_C^A$ and $\text{dec}_C^B$, which determine whether $A$ and $B$ obtain a session key or not.

**Ideal model.** Let $A, B$ be the honest parties and let $C_{\text{ideal}}$ be any PPT ideal adversary with auxiliary input $\sigma$.

1. $A$ and $B$ receive $w \overset{\text{R}}{\leftarrow} \mathcal{D}$.
2. $A$ and $B$ both send $w$ to the trusted party.
3. $C_{\text{ideal}}$ sends $(\text{dec}_C^A, \text{dec}_C^B)$ to the trusted party.
4. The trusted party chooses $K \overset{\text{R}}{\leftarrow} \{0,1\}^n$. For each party $i \in \{A, B\}$, the trusted party sends $K$ if $\text{dec}_C^i = 1$ and sends $\perp$ if $\text{dec}_C^i = 0$.

The ideal distribution is defined by:

$$\text{IDEAL}_{C_{\text{ideal}}}(\mathcal{D}, \sigma) = (w, \text{output}(A), \text{output}(B), \text{output}(C_{\text{ideal}}(\sigma))) \ .$$

**Real model.** Let $A, B$ be the honest parties and let $C$ be any PPT real adversary with auxiliary input $\sigma$.

At some initialization stage, $A$ and $B$ receive $w \overset{\text{R}}{\leftarrow} \mathcal{D}$. The real protocol is executed by $A$ and $B$ communicating via $C$. We will augment $C$'s view of the protocol with $A$ and $B$'s decision bits, denoted by $\text{dec}_A$ and $\text{dec}_B$, where $\text{dec}_A = \texttt{reject}$ if $\text{output}(A) = \perp$, and $\text{dec}_A = \texttt{accept}$ otherwise ($\text{dec}_B$ is defined similarly). (Indeed, in typical applications, the decisions of $A$ and $B$ will be learned by the real adversary $C$: if $A$ obtains a session key, then it will use it afterwards; otherwise, $A$ will stop communication or try to re-initiate an execution of the protocol.) $C$'s augmented view is denoted by $\text{output}(C^{A(w),B(w)}(\sigma))$.

The real distribution is defined by:

$$\text{REAL}_C(\mathcal{D}, \sigma) = (w, \text{output}(A), \text{output}(B), \text{output}(C^{A(w),B(w)}(\sigma))) \ .$$

One might want to say that a protocol for password-based session-key generation is secure if the above ideal and real distributions are computationally indistinguishable. Unfortunately, as pointed in [12], an active adversary can guess the password and successfully impersonate one of the parties with probability $\frac{1}{|\mathcal{D}|}$. This implies that the real and ideal distributions are always distinguishable with probability at least $\frac{1}{|\mathcal{D}|}$. Thus we will only require that the distributions be distinguishable with probability at most $O(\gamma)$ where the goal is to make $\gamma$ as close to $\frac{1}{|\mathcal{D}|}$ as possible. In the case of a passive adversary, we require that the real and ideal distributions be computationally indistinguishable (for all subsequent definitions, this requirement will be implicit).

**Definition 1 (Initial definition).** *A protocol for password-based authenticated session-key generation is* $(1-\gamma)$-secure *for the dictionary* $\mathcal{D} \subseteq \{0,1\}^n$ *(where* $\gamma$ *is a function of the dictionary size* $|\mathcal{D}|$ *and* $n$*) if:*

1. *For every real* passive *adversary, there exists an ideal adversary* $C_{\text{ideal}}$ *which always sends (1,1) to the trusted party such that for every auxiliary input* $\sigma \in \{0,1\}^{\text{poly}(n)}$*,*

$$\{\text{IDEAL}_{C_{\text{ideal}}}(\mathcal{D}, \sigma)\}_\sigma \overset{\text{c}}{\equiv} \{\text{REAL}_C(\mathcal{D}, \sigma)\}_\sigma \ .$$

2. *For every real adversary* $C$*, there exists an ideal adversary* $C_{\text{ideal}}$ *such that for every auxiliary input* $\sigma \in \{0,1\}^{\text{poly}(n)}$*,*

$$\{\text{IDEAL}_{C_{\text{ideal}}}(\mathcal{D}, \sigma)\}_\sigma \overset{\text{O}(\gamma)}{\equiv} \{\text{REAL}_C(\mathcal{D}, \sigma)\}_\sigma \ .$$

By the discussion above, the best we can hope for is $\gamma = \frac{1}{|\mathcal{D}|}$. Note that in [12], their definition and protocol refer to any dictionary $\mathcal{D} \subseteq \{0,1\}^n$ and $\gamma = \frac{1}{|\mathcal{D}|}$. In contrast, our protocol will be $(1-\gamma)$-secure for dictionaries of the form $\mathcal{D} = \{0,1\}^d$ and $\gamma = \left(\frac{\text{poly}(n)}{|\mathcal{D}|}\right)^{\Omega(1)}$.

## 2.2 Augmented Definitions

The above definition is actually not completely satisfying because of a subtle point raised by Rackoff: the adversary controls the scheduling of the interactions $(A, C)$ and $(C, B)$ so the honest parties do not necessarily end at the same time. $A$ might use its session key $K_A$ before the interaction $(C, B)$ is completed: $A$'s use of $K_A$ leaks information which $C$ might use in its interaction with $B$ to learn $K_A, K_B$ or the password $w$.

In [12], Goldreich and Lindell augment the above definition with a *session-key challenge* to address this issue. Suppose that $A$ completes the protocol first and outputs a session key $K$, then the adversary is given a session-key challenge $K_\beta$, which is the session key $K$ with probability 1/2 (i.e. $\beta = 1$) or a truly random string $K_0$ with probability 1/2 (i.e. $\beta = 0$). The adversary $C$ will be given the session-key challenge in both the ideal and real models, as soon as the first honest party outputs a session-key $K$. We call the resulting definition *security with respect to the session-key challenge.*

Goldreich and Lindell give some intuition as to why the session-key challenge solves the above flaw. First, note that the ideal adversary cannot distinguish between the case $\beta = 0$ and the case $\beta = 1$ since in the ideal model, both $K_0$ and $K$ are truly uniform strings. Consider the real adversary who has been given the session-key challenge: if $C$ has been given $K_0$, then the session-key challenge does not help $C$ in attacking the protocol, since $C$ could generate $K_0$ on its own. Suppose that instead $C$ has been given $K$ and can somehow use it to attack the protocol (this corresponds to the situation where $A$ uses the session key $K$; $C(K)$ can simulate $A$'s use of the key), then it would mean that $C$ can tell if it is in the case $\beta = 0$ or $\beta = 1$, which is not possible if the protocol is secure with respect to the session-key challenge.

Our intuitive notion of security is that *no matter how $A$ uses its session key $K$* before the execution $(C, B)$ is completed, the ideal and real distributions should be $(1 - O(\gamma))$-indistinguishable. Even with the above intuition, it is not immediate that the session-key challenge fully captures this goal. Thus we propose an alternative augmentation to Definition 1 that corresponds more directly to this goal.

We model the different ways the party $A$ could use its session key $K$ by considering an arbitrary probabilistic polynomial time machine $Z$ which is given the key $K$ (as soon as $A$ outputs a session key $K$) and interacts with the adversary in both the ideal and real models. This is similar to the "application" queries in Shoup's model for (non-password-based) secure key exchange [22], which was later extended to password protocols in [7]. $Z$ can also be thought of in terms of "environment" as in the definition of universal composability by Canetti [8]: $Z$ models an arbitrary environment (or application) in which the key generated by the session-key generation protocol is used.[2]

Examples of environments follow:

1. $Z(K) = \perp$: $A$ does not use its session key.
2. $Z(K) = K$: $A$ publicly outputs its session key.
3. $Z(K) = K$ with probability 1/2, $U_n$ with probability 1/2. This corresponds to the session-key challenge.
4. $Z(K) = \text{Enc}_K(0^n)$: $A$ uses its session-key for secure private-key encryption.
5. $C$ sends a query $m_1$, $Z(K)$ answers with $\text{Enc}_K(m_1)$, $C$ sends a query $m_2$, $Z(K)$ answers with $\text{Enc}_K(m_2)$ and so on. $A$ uses its key for encryption and the adversary is mounting a chosen plaintext attack.

We call the definition obtained by adding (in both the ideal and real models) the environment $Z$ *security with respect to the environment.* Informally, a real protocol is secure with respect to the environment if every adversary attacking the real protocol and interacting with an arbitrary environment can be simulated, with probability $1 - O(\gamma)$, by an ideal adversary attacking the ideal functionality and interacting with the same environment in the ideal model. (More precisely, for every real adversary, there should be a *single* ideal adversary that simulates it well for *every* environment.)

Note that security with respect to the environment implies security with respect to the session-key challenge since it suffices to consider the PPT $Z(K)$ which generates $\beta \overset{\text{R}}{\leftarrow} \{0, 1\}$ and outputs the key $K$ if $\beta = 1$ or a truly random string $K_0$ if $\beta = 0$. We show that the two definitions are actually equivalent:

**Theorem 2.** *A protocol $(A, B)$ is $(1 - \gamma)$-secure with respect to the session-key challenge iff it is $(1 - \gamma)$-secure with respect to the environment.*

This is similar to a result of Shoup [22] showing the equivalence of his definition and the Bellare-Rogaway [3] definition for non-password-based key exchange. The "application" queries in Shoup's definition are analogous to our

[2] Note that this is not as general as the definition of Canetti since the environment $Z$ is only given the session key and not the password $w$.

environment $Z$, and the “test” queries in [3] are analogous to the session-key challenge. Though both of these definitions have been extended to password-authenticated key exchange [7,2], it is not immediate that Shoup’s equivalence result extends directly to our setting. For example, the definitions of [3,2] are not simulation-based and do not directly require that the password remain pseudorandom, whereas here we are relating two simulation-based definitions that do ensure the password’s secrecy.

Given Theorem 2, the relationship between security with respect to the environment and security with respect to the session-key challenge is analogous to the relationship between semantic security and indistinguishability for encryption schemes [14,18]. Though both are equivalent, the former captures our intuitive notion of security better, but the latter is typically easier to establish for a given protocol (as it involves only taking into account a specific environment $Z$).

## 3 An Overview of the Protocol

Before presenting our protocol, we introduce the polynomial evaluation functionality, which is an important tool for the rest of the paper. In [19], it is observed that a secure protocol for polynomial evaluation immediately yields a protocol for session-key generation which is secure against *passive* adversaries. In [12], Goldreich and Lindell work from the intuition (from [6]) that by augmenting a secure protocol for polynomial evaluation with additional mechanisms, one can obtain a protocol for session-key generation which is secure against *active* adversaries. Our protocol also comes from this intuition but the additional tools we are using are different.

### 3.1 Secure Polynomial Evaluation

In a secure polynomial evaluation, a party $A$ knows a polynomial $Q$ over some field $\mathbb{F}$ and a party $B$ wishes to learn the value $Q(x)$ for some element $x \in \mathbb{F}$ such that $A$ learns nothing about $x$ and $B$ learns nothing else about the polynomial $Q$ but the value $Q(x)$. More specifically, for our problem, we will assume that $\mathbb{F} = \mathrm{GF}(2^n) \approx \{0,1\}^n$, $Q$ is a linear non-constant polynomial over $\mathbb{F}$, and $x$ is a string in $\{0,1\}^n$.

**Definition 3 (Polynomial evaluation).** *The polynomial evaluation functionality is defined as:*

**Inputs** *The input of $A$ is a linear non-constant polynomial $Q$ over $\mathrm{GF}(2^n)$. The input of $B$ is a value $x \in \mathrm{GF}(2^n)$.*
**Outputs** *$B$ receives $Q(x)$. $A$ receives nothing.*

As observed in [19], a secure protocol for polynomial evaluation yields immediately a protocol for session-key generation which is secure against passive adversaries as follows: $A$ chooses a random linear non-constant polynomial $Q$, and $A$ and $B$ engage in a secure polynomial evaluation protocol, where $A$ inputs $Q$ and $B$ inputs $w$, so that $B$ obtains $Q(w)$. Since $A$ has both $Q$ and $w$, $A$ can also obtain $Q(w)$, and the session key is set to be $K = Q(w)$.

This protocol is secure against passive adversaries because the key $K$ is a random string (since $Q$ is a random polynomial), and it can be shown that an eavesdropper learns nothing about $w$ or $Q(w)$ (due to the security of the polynomial evaluation).

However, the protocol is not secure against active adversaries. For example, an active adversary $C$ can input a fixed polynomial $Q_C$ in its interaction with $B$, say the identity polynomial $id$, and a fixed password $w_C$ in its interaction with $A$. $A$ outputs the session key $Q_A(w)$ and $B$ outputs the session key $Q_C(w) = w$. With probability $1-2^{-n}$, the two session keys are different, whereas the definition of security requires them to be equal with probability $1 - O(\gamma)$.

## 3.2 Motivation for Our Protocol

The main deficiency of the secure polynomial evaluation protocol against active adversaries is that it does not guarantee that $A$ and $B$ output the same random session key. Somehow, the parties have to check that they computed the same random session key before starting to use it. It can be shown that $A$'s session key $K_A = Q_A(w)$ is pseudorandom to the adversary, so $A$ can start using it without leaking information. However, $B$ cannot use its key $K_B = Q_C(w)$ because it might belong to a set of polynomial size (for example, if $Q_C = id$, then $Q_C(w) \in \mathcal{D}$ where the dictionary is by definition a small set). Hence Goldreich and Lindell added a validation phase in which $A$ sends a message to $B$ so that $B$ can check if it computed the same session key, say $A$ sends $f^n(K_A)$ where $f$ is a one-way permutation. Since $f^n$ is a 1-1 map, this uniquely defines $K_A$ (the session key used now consists of hardcore bits of $f^i(K_A)$, for $i = 0, \cdots, n-1$) : $B$ will compute $f^n(K_B)$ and compare it with the value it received.

But it is still not clear that this candidate protocol is secure. Recall that the security of the polynomial evaluation protocol applies only in the stand-alone setting and guarantees nothing in the concurrent setting. In particular, it might be that $C$ inputs a polynomial $Q_C$ in the polynomial evaluation between $C$ and $B$ such that the polynomials $Q_A$ and $Q_C$ are related in some manner, say for any $w \in \mathcal{D}$, it is easy to compute the correct validation message $f^{2n}(Q_C(w))$ given the value of $f^{2n}(Q_A(w))$; yet $B$'s key does not equal $A$'s key.

To prevent this from happening, Goldreich and Lindell force the polynomial $Q$ input in the polynomial evaluation phase to be consistent with the message sent in the validation phase (which is supposedly $f^{2n}(Q(w))$). The parties have to commit to their inputs at the beginning and then prove in a zero-knowledge manner that the messages sent in the validation phase are consistent with these commitments. Because of the person-in-the-middle attack and the concurrency issues mentioned earlier, Goldreich and Lindell cannot use standard commitment schemes and standard zero-knowledge proofs but rather they use nonmalleable commitments and the specific zero-knowledge proof of Richardson and Kilian.

Our approach is to allow $C$ to input a polynomial $Q_C$ related to $Q_A$, but to prevent $C$ from being able to compute a correct validation message with respect to $B$'s session-key, even given $A$'s validation message. Suppose that the parties have access to a family of pairwise independent hash functions $\mathcal{H}$. In the validation phase, we require $A$ to send $h(f^{2n}(K_A)) = h(f^{2n}(Q_A(w)))$ for some function $h \overset{\text{R}}{\leftarrow} \mathcal{H}$. Then, even if $K_A = Q_A(w)$ and $K_B = Q_C(w)$ are related (but distinct), the values $h(f^{2n}(K_A))$ and $h(f^{2n}(K_B))$ will be independent and $C$ cannot do much better than randomly guess the value of $h(f^{2n}(K_B))$.

One difficulty arises at this point: the parties have to agree on a common random hash function $h \overset{\text{R}}{\leftarrow} \mathcal{H}$. But the honest parties $A$ and $B$ only share the randomness coming from the password $w$ so this password $w$ has to be enough to agree on a random hash function. To make this possible, we assume that the password is the form $(w, w')$ where $w$ and $w'$ are chosen independently of one another: $w$ is chosen at random from an arbitrary dictionary $\mathcal{D} \subseteq \{0,1\}^n$ and $w'$ is uniformly distributed in $\mathcal{D}' = \{0,1\}^{d'}$. (For example, these can be obtained by splitting a single random password from $\{0,1\}^{d''}$ into two parts.) The first part of the password, $w$, will be used in the polynomial evaluation protocol whereas the second part of the password, $w'$, will be used as the index of a hash function. Indeed, if we assume that $\mathcal{D}' = \{0,1\}^{d'}$, there exists a family of almost pairwise independent hash functions $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$, where each hash function is indexed by a password $w' \in \mathcal{D}'$ and $m = \Omega(d')$.

We formalize these ideas in the protocol described below.

## 3.3 Description of the Protocol

Like in [12], we will need a secure protocol for an augmented version of polynomial evaluation.

**Definition 4 (Augmented polynomial evaluation).** *The augmented polynomial evaluation functionality is defined as:*

**Earlier phase.** *$A$ sends a commitment $c_A = \mathrm{Commit}(Q_A, r_A)$ to a linear non-constant polynomial $Q_A$ for a randomly chosen $r_A$. $B$ receives a commitment $c_B$. We assume that the commitment scheme used is perfectly binding and computationally hiding.*

**Inputs.** *The input of $A$ is a linear non-constant polynomial $Q_A$, a commitment $c_A$ to $Q_A$ and a corresponding decommitment $r_A$. The input of $B$ is a commitment $c_B$ and a value $x \in \mathrm{GF}(2^n)$.*

**Outputs.**
- *In the case of correct inputs, i.e. $c_A = c_B$ and $c_A = \mathrm{Commit}(Q_A, r_A)$, $B$ receives $Q_A(x)$ and $A$ receives nothing.*
- *In the case of incorrect inputs, i.e. $c_A \neq c_B$ or $c_A \neq \mathrm{Commit}(Q_A, r_A)$, $B$ receives a special failure symbol $\perp$ and $A$ receives nothing.*

The other cryptographic tools we will need are:

**Commitment scheme:** Let Commit be a perfectly binding, computationally hiding string commitment.

**Seed-committed pseudorandom generator:** Similarly to [12], we will use the seed-committed pseudorandom generator

$$G(s) = (b(s)b(f(s)) \cdots b(f^{n+\ell-1}(s))f^{n+\ell}(s))$$

where $f$ is a one-way permutation with hardcore bit $b$.

**One-time MAC with pseudorandomness property:** Let MAC be a message authentication code for message space $\{0,1\}^{p(n)}$ (for a polynomial $p(n)$ to be specified later) using keys of length $\ell = \ell(n)$ that is secure against one query attack, i.e. a PPT $A$ which queries the tagging algorithm $\mathrm{MAC}_K$ on *at most one* message of its choice cannot produce a valid forgery on a different message. Additionally, we will require the following pseudorandomness property:
- Let $K$ be a uniform key of length $\ell$.
- The adversary queries the tagging algorithm $\mathrm{MAC}_K$ on the message $m$ of its choice.
- The adversary selects $m' \neq m$. We require that the value $\mathrm{MAC}_K(m')$ be pseudorandom with respect to the adversary's view.

Two examples of such a MAC are:
- $\mathrm{MAC}_s(m) = f_s(m)$ where $\{f_s\}_{s\in\{0,1\}^\ell}$ is a pseudorandom function family
- $\mathrm{MAC}_{a,b}(m) = am + b$ where $\ell(n) = 2p(n)$ and $a, b \in \mathrm{GF}(2^{\ell/2})$.

**Almost pairwise independent hash functions:** The family of functions $\mathcal{H} = \{h_{w'} : \{0,1\}^n \to \{0,1\}^m\}_{w'\in\{0,1\}^{d'}}$ is said to be *pairwise $\delta$-dependent* or *almost pairwise independent* if:
1. (uniformity) $\forall x \in \{0,1\}^n$, when we choose $w' \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{d'}$, $h_{w'}(x)$ is uniform over $\{0,1\}^m$.
2. (pairwise independence) $\forall x_1 \neq x_2 \in \{0,1\}^n, \forall y_1, y_2 \in \{0,1\}^m$, when we choose $w' \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{d'}$,

$$\Pr_{w'\in\{0,1\}^{d'}}[h_{w'}(x_1) = y_1 \wedge h_{w'}(x_2) = y_2] = \frac{1+\delta}{2^{2m}} .$$

We also require that for a fixed $w' \in \{0,1\}^{d'}$, the function $h_{w'}$ is regular, i.e. it is $2^{n-m}$ to 1. In other words, $h_{w'}(U_n) \equiv U_m$. Throughout this paper, we write $\mu \overset{\mathrm{def}}{=} \frac{1+\delta}{2^m}$.

**Lemma 5.** *For the fixed dictionary $\mathcal{D}' = \{0,1\}^{d'} \subseteq \{0,1\}^n$, there exists a family of almost pairwise independent hash functions $\mathcal{H} = \{h_{w'} : \{0,1\}^n \to \{0,1\}^m\}_{w'\in\mathcal{D}'}$ for $\mu = O\left(\frac{n}{|\mathcal{D}'|^{1/3}}\right)$.*

The formal description of the protocol follows. A schematic diagram of the protocol is given in Fig. 1.

**Protocol 6.** 1. **Inputs:** The parties $A$ and $B$ have a joint password $(w, w')$ where $w$ and $w'$ are chosen independently: $w$ is chosen at random from an arbitrary dictionary $\mathcal{D} \subseteq \{0,1\}^n$ and $w'$ is uniformly distributed in $\mathcal{D}' = \{0,1\}^{d'} \subseteq \{0,1\}^n$.

2. **Commitment:** $A$ chooses a random linear non-constant polynomial $Q_A$ over $\mathrm{GF}(2^n)$ and coin tosses $r_A$ and sends $c_A = \mathrm{Commit}(Q_A, r_A)$. $B$ receives some commitment $c_B$.
3. **Augmented polynomial evaluation:**
   a) $A$ and $B$ engage in a polynomial evaluation protocol: $A$ inputs the polynomial $Q_A$, the commitment $c_A$ and the coin tosses $r_A$ it used for the commitment; $B$ inputs the commitment $c_B$ it received and the password $w$ seen as an element of $GF(2^n)$.
   b) The output of $B$ is denoted $\Pi_B$, which is supposed to be equal to $Q_A(w)$.
   c) $A$ internally computes $\Pi_A = Q_A(w)$.
4. **Validation:**
   a) $A$ sends the string $y_A = h_{w'}(f^{n+\ell}(\Pi_A))$.
   b) Let $t_A$ be the session transcript so far as seen by $A$. $A$ computes $k_1(\Pi_A) = b(\Pi_A) \cdots b(f^{\ell-1}(\Pi_A))$ and sends the string $z_A = \mathrm{MAC}_{k_1(\Pi_A)}(t_A)$.
5. **Decision:**
   a) $A$ always accepts and outputs $k_2(\Pi_A) = b(f^{\ell}(\Pi_A)) \cdots b(f^{\ell+n-1}(\Pi_A))$
   b) $B$ accepts (this event is denoted by $\mathrm{dec}_B = \mathtt{accept}$) if the strings $y_B$ and $z_B$ it received satisfy the following conditions :
      – $y_B = h_{w'}(f^{n+\ell}(\Pi_B))$
      – $\mathrm{Ver}_{k_1(\Pi_B)}(t_B, z_B) = \mathtt{accept}$, where $t_B$ is the session transcript so far as seen by $B$ and $k_1(\Pi_B)$ is defined analogously to $k_1(\Pi_A)$.

      If $\Pi_B = \perp$, then $B$ will immediately reject. If $B$ accepts, it outputs $k_2(\Pi_B) = b(f^{\ell}(\Pi_B)) \cdots b(f^{\ell+n-1}(\Pi_B))$.

## 4 Security Theorems

**Theorem 7.** *Protocol 6 is secure for the dictionary $\mathcal{D} \times \mathcal{D}' = \mathcal{D} \times \{0,1\}^{d'}$ against passive adversaries. More formally, for every passive PPT real adversary $C$, there exists an ideal adversary $C_{\mathrm{ideal}}$ which always sends $(\mathrm{dec}_C^A, \mathrm{dec}_C^B) = (1,1)$ to the trusted party such that for every auxiliary input $\sigma \in \{0,1\}^{\mathrm{poly}(n)}$:*

$$\{\mathrm{IDEAL}_{C_{\mathrm{ideal}}}(\mathcal{D} \times \mathcal{D}', \sigma)\}_\sigma \overset{\mathrm{c}}{\equiv} \{\mathrm{REAL}_C(\mathcal{D} \times \mathcal{D}', \sigma)\}_\sigma \ .$$

**Theorem 8.** *Protocol 6 is $(1-\gamma)$-secure with respect to the session-key challenge for the dictionary $\mathcal{D} \times \mathcal{D}' = \mathcal{D} \times \{0,1\}^{d'}$, for $\gamma = \max\left\{\frac{1}{|\mathcal{D}|}, \left(\frac{\mathrm{poly}(n)}{|\mathcal{D}'|}\right)^{\Omega(1)}\right\}$. More precisely, $\gamma = \max\left\{\frac{1}{|\mathcal{D}|}, O\left(\left(\frac{n^3}{|\mathcal{D}'|}\right)^{1/6}\right)\right\}$.*

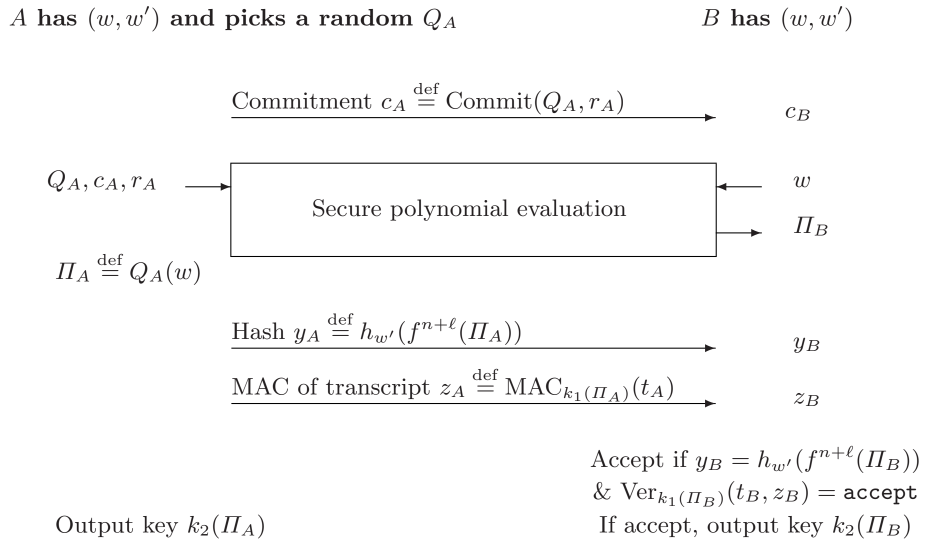
$A$ **has** $(w, w')$ **and picks a random** $Q_A$
$B$ **has** $(w, w')$
Commitment $c_A \overset{\text{def}}{=} \text{Commit}(Q_A, r_A)$
$c_B$
$Q_A, c_A, r_A$
Secure polynomial evaluation
$w$
$\Pi_B$
$\Pi_A \overset{\text{def}}{=} Q_A(w)$
Hash $y_A \overset{\text{def}}{=} h_{w'}(f^{n+\ell}(\Pi_A))$
$y_B$
MAC of transcript $z_A \overset{\text{def}}{=} \text{MAC}_{k_1(\Pi_A)}(t_A)$
$z_B$
Accept if $y_B = h_{w'}(f^{n+\ell}(\Pi_B))$
& $\text{Ver}_{k_1(\Pi_B)}(t_B, z_B) = \texttt{accept}$
Output key $k_2(\Pi_A)$
If accept, output key $k_2(\Pi_B)$

**Fig. 1.** Overview of our protocol

The shared dictionary of the form $\mathcal{D} \times \{0,1\}^d$ required in Theorem 8 can be realized from several other types of dictionaries $\mathcal{D}''$, achieving security bounds of the form $(\text{poly}(n)/|\mathcal{D}''|)^{\Omega(1)}$ in all cases:

**Single Random Password**
We can split a single random password from a dictionary $\mathcal{D}'' = \{0,1\}^{d''}$ into two parts, one of length $d$ and one of length $d'$.

**Arbitrary Password with Common Random String**
We can convert a password from an arbitrary dictionary $\mathcal{D}'' \subseteq \{0,1\}^n$ into a single random password (as in the previous bullet) in the common random string model. Specifically, we view the common random string $r \in \{0,1\}^\ell$ as the seed for a *randomness extractor* $\text{Ext} : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^{d''}$ [20]. Given password $w \leftarrow \mathcal{D}''$, the honest parties can compute an (almost-uniform) password $\text{Ext}(w, r)$. Using the low min-entropy extractors of [13,23], the length of the common random string need only be $\ell = O(\log n + \log |\mathcal{D}''|)$. (Unlike the protocols of [12] and [16], this requires knowing a lower bound on the size of the dictionary $|\mathcal{D}''|$.)

**Two Independent Passwords**
If the parties share *two* independent passwords $w_1, w_2$ coming from arbitrary dictionaries $\mathcal{D}_1'', \mathcal{D}_2'' \subseteq \{0,1\}^n$, then they can apply an extractor for 2 independent weak random sources [9] to convert these into an almost-uniform password. Unfortunately *explicit* constructions for 2-source extractors are only known when $|\mathcal{D}_1''| \cdot |\mathcal{D}_2''| \geq 2^n$, but nonconstructively there exist 2-source extractors that would only require the dictionaries to be of size $\text{poly}(n)$.

## 5 Overview of the Proof

Like in [12], the main part of the proof of Theorem 8 is the *key-match property*: if $\Pi_A \neq \Pi_B$, then $B$ will reject with probability $1 - O(\gamma)$. Once the key-match property is established, we can easily adapt the proofs in [12] to our specific protocol to build an ideal adversary which simulates the real adversary's view.

The main part of our proof that is new (and simpler than [12]) is the key-match property. As noted in the introduction, the adversary $C$ has total control over the scheduling of the two interactions $(A, C)$ and $(C, B)$. Hence the key-match property will be proved for every possible scheduling case, including those for which these interactions are concurrent. Nevertheless, the key-match property will be established by tools of secure two-party computation, which *a priori* only guarantee security in the stand-alone setting.

Recall that $B$ accepts iff two conditions are satisfied: the string $y_B$ received must equal $h_{w'}(f^{n+\ell}(\Pi_B))$ and the MAC $z_B$ received must be a valid MAC, i.e. $\mathrm{Ver}_{k_1(\Pi_B)}(t_B, z_B) = \texttt{accept}$. Hence, to establish the key-match property, we can omit the verification of the MAC by $B$ and only consider the probability that $C$ succeeds in sending the value $h_{w'}(f^{n+\ell}(\Pi_B))$ when $\Pi_A \neq \Pi_B$. (Like in [12], the MAC is only used to reduce the simulation of active adversaries to the simulation of passive adversaries plus the key-match property.)

We consider two scheduling cases (see Figures 2 and 3):

**Scheduling 1** : $C$ sends the commitment $c_B$ to $B$ *after* $A$ sends the hash value $y_A$.
The intuition for this case is that we have two sequential executions $(A, C)$ and $(C, B)$. Using the security of the polynomial evaluation $(A, C)$, we show that even if $C$ receives $y_A$, the hash index $w'$ is $(1 - \epsilon)$ pseudorandom with respect to the adversary's view. Hence, by the uniformity property of the hash functions, $C$ cannot do much better than randomly guess the value of $h_{w'}(f^{n+\ell}(\Pi_B))$.

**Scheduling 2** : $C$ sends the commitment $c_B$ to $B$ *before* $A$ sends the hash value $y_A$.
The almost pairwise independence property means that for fixed values $x_1 \neq x_2 \in \{0,1\}^n$, if the index $w'$ is chosen at random and independently of $x_1$ and $x_2$, then given the value $h_{w'}(x_1)$, one cannot do much better than randomly guess the value $h_{w'}(x_2)$. Before $y_A$ is sent, the hash index $w'$ is random (since it has not been used by $A$). Thus, if we show that the values $\Pi_A$ and $\Pi_B$ can be computed before $y_A$ is sent, then $w'$ is independent of $x_1 = f^{n+\ell}(\Pi_A)$ and $x_2 = f^{n+\ell}(\Pi_B)$ and the adversary cannot guess $h_{w'}(x_2)$ even given $y_A = h_{w'}(x_1)$. To show that $\Pi_A$ and $\Pi_B$ can be computed before $y_A$ is sent, we use an *ideal* augmented polynomial evaluation $(C, B)$ to extract an opening of the adversary's commitment $c_B$. (The adversary must input such an opening in the ideal evaluation, else $B$ will reject.)

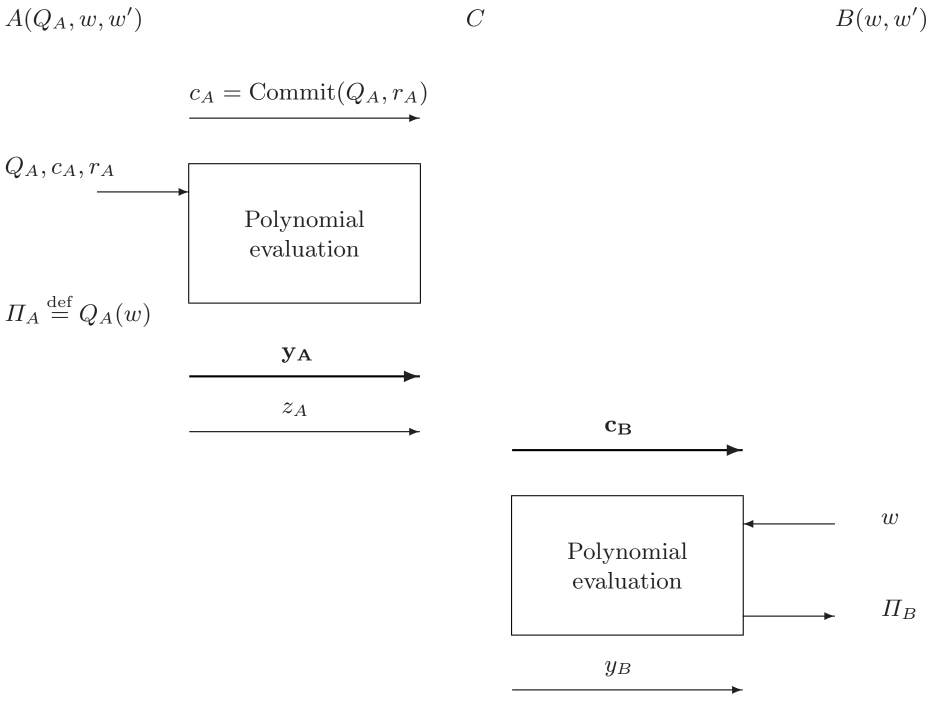

**Fig. 2.** First scheduling

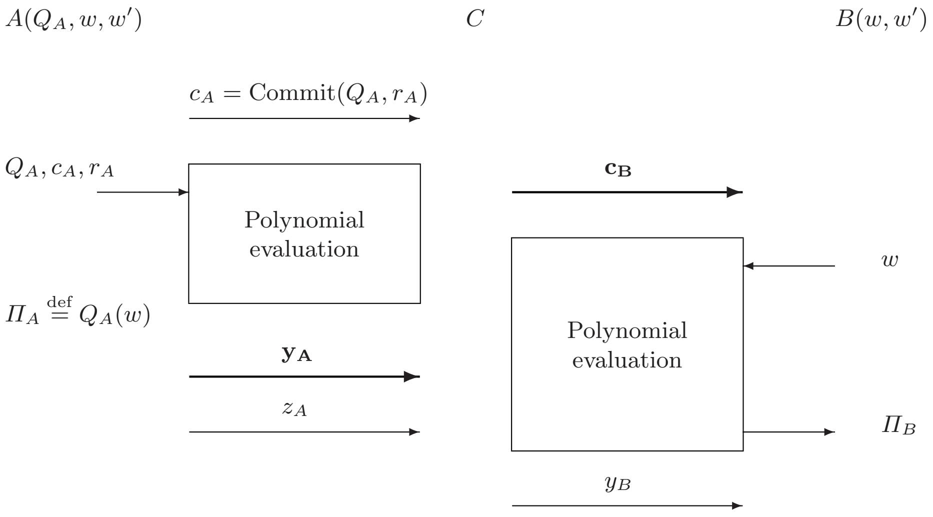

**Fig. 3.** Second scheduling

**Acknowledgments.** We thank Oded Goldreich and Yehuda Lindell for their encouragement and an inspiring discussion which led to a substantial simplification of our protocol. We are also grateful to Mihir Bellare for pointing out the extension of our protocol to arbitrary dictionaries in the common random string model.

## References

1. Barak, B.: Constant-Round Coin-Tossing With a Man in the Middle or Realizing the Shared Random String Model. IEEE Symposium on Foundations of Computer Science (2002) 345–355
2. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. Advances in Cryptology–Eurocrypt 2000 Proceedings, Lecture Notes in Computer Science **1807** (2000) 139–155
3. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. Advances in Cryptology - Crypto 93 Proceedings, Lecture Notes in Computer Science **773** (1994) 232–249
4. Bellovin, S., Merritt, M.: Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks. ACM/IEEE Symposium on Research in Security and Privacy (1992) 72–84
5. Bellovin, S., Merritt, M.: Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise. ACM Conference on Computer and Communications Security (1993) 244–250
6. Boyarsky, M.: Public-Key Cryptography and Password Protocols: The Multi-User Case. ACM Conference on Computer and Communications Security (1999) 63–72
7. Boyko, V., MacKenzie, P., Patel, S.: Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. Advances in Cryptology - Eurocrypt 2000 Proceedings, Lecture Notes in Computer Science **1807** (2000) 156–171
8. Canetti, R.: Universally Composable Security: A New Paradigm for Cryptographic Protocols. IEEE Symposium on Foundations of Computer Science (2001) 136–145
9. Chor, B., Goldreich, O.: Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. SIAM Journal on Computing **17:2** (1988) 230–261
10. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory **22:6** (1976) 644–654
11. Gennaro, R., Lindell, Y.: A Framework for Password-Based Authenticated Key Exchange. Advances in Cryptology - Eurocrypt 2003 Proceedings, Lecture Notes in Computer Science **2656** (2003) 524–543
12. Goldreich, O., Lindell, Y.: Session-Key Generation Using Human Passwords Only. Advances in Cryptology - Crypto 2001 Proceedings, Lecture Notes in Computer Science **2139** (2001) 408–432
13. Goldreich, O., Wigderson, A.: Tiny Families of Functions with Random Properties: A Quality-Size Trade-off for Hashing. Random Structures and Algorithms **11:4** (1997) 315–343
14. Goldwasser, S., Micali, S.: Probabilistic Encryption. Journal of Computer and System Sciences **28:2** (1984) 270–299
15. Halevi, S., Krawczyk, H.: Public-Key Cryptography and Password Protocols. ACM Conference on Computer and Communications Security (1998) 122–131

16. Katz, J., Ostrovsky, R., Yung, M.: Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. Advances in Cryptology - Eurocrypt 2001 Proceedings, Lecture Notes in Computer Science **2045** (2001) 475–494
17. Kobara, K., Imai, H.: Pretty-Simple Password-Authenticated Key-Exchange Under Standard Assumptions. IECIE Trans. **E85-A:10** (2002) 2229–2237
18. Micali, S., Rackoff, C., Sloan, B.: The Notion of Security for Probabilistic Cryptosystems. SIAM Journal on Computing **17** (1988) 412–426
19. Naor, M., Pinkas, B.: Oblivious Transfer and Polynomial Evaluation. ACM Symposium on Theory of Computing. (1999) 245–254
20. Nisan, N., Zuckerman, D.: Randomness is Linear in Space. Journal of Computer and System Sciences **52:1** (1996) 43–52
21. Richardson, R., Kilian, J.: On the Concurrent Composition of Zero-Knowledge Proofs. Advances in Cryptology - Eurocrypt 99 Proceedings, Lecture Notes in Computer Science **1592** (1999) 415–431
22. Shoup, V.: On Formal Models for Secure Key Exchange. Cryptology ePrint Archive (1999) Report 1999/012
23. Srinivasan, A., Zuckerman, D.: Computing with Very Weak Random Sources. IEEE Symposium on Foundations of Computer Science (1994) 264–275
24. Steiner, M., Tsudik, G., Waidner, M.: Refinement and Extension of Encrypted Key Exchange. Operating Systems Review **29:3** (1995) 22–30