# A New Privacy-Enhanced Matchmaking Protocol

Ji Sun Shin
Dept. of Computer Science
University of Maryland
College Park, MD 20740
sunny@cs.umd.edu

Virgil D. Gligor
Dept. of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
gligor@cmu.edu

## Abstract

*Although several wide-spread internet applications (e.g., job-referral services, dating services) can benefit from online matchmaking, protocols defined over the past two decades fail to address important privacy concerns. In this paper, we enhance traditional privacy requirements (e.g., user anonymity, matching-wish authenticity) with new privacy goals (e.g., resistance to off-line dictionary attacks, and forward privacy of users' identities and matching wishes), and argue that privacy-enhanced matchmaking cannot be provided by solutions to seemingly related problems such as secret handshakes, set intersection, and trust negotiation. We define an adversary model, which captures the key security properties of privacy-enhanced matchmaking, and show that a simple, practical protocol derived by a two-step transformation of a password-based authenticated key exchange counters adversary attacks in a provable manner (in the standard model of cryptographic security).*

## 1 Introduction

The notion of on-line matchmaking was introduced by Baldwin and Gramlich in 1985 [4] anticipating a now-common type of internet service: a job-referral service matching a company that wants to hire an employee having certain characteristics or "wishes" (e.g., skills, level of experience, salary level, credentials), without advertizing the job publicly, with an applicant who wants to find a new job without revealing her/his plan to leave the current job. Currently, many other internet services require different forms of on-line matchmaking capabilities; e.g., internet dating services. Further, applications of mobile ad-hoc networks, such as dynamic discovery of peer nodes with identical characteristics (e.g., provenance, configuration, capabilities, level of trust) may benefit from private matchmaking capabilities.

Baldwin and Gramlich provided a solution for on-line matchmaking intended to support (1) anonymity of users (i.e., protecting company and job seekers' identities), (2) authentication of matches, and (3) joint notification to users only in the event of a positive match (i.e., a job seeker's identity is authenticated to the company and vice-versa only when the job-seeker's wishes match the company job requirements). Their solution required a *trusted* matchmaker who learned the identities of the protocol users and their wishes and was relied upon not to reveal them. Analysis of the Baldwin-Gramlich protocol shows that their solution can be broken via a simple message replacement attack (viz., Zhang and Needham's attack [58]) revealing users' identities and their desired characteristics to an adversary.

In 1986, Meadows introduced a correct protocol for online matchmaking that is independent of an on-line trusted matchmaker [47] beyond the initialization step. Although Meadows' solution provides privacy for the users' credentials, it does not aim at providing anonymity of protocol users, and hence cannot be used for *private* matchmaking.

More recently, Zhang and Needham [58] developed a simple protocol for on-line matchmaking providing some degree of user anonymity and privacy of matching wishes. Their protocol removes any direct interaction between users and requires an untrusted on-line matchmaker that acts as a public bulletin-board which posts encrypted user wishes for retrieval by all interested users. Specifically, a user $U$ hashes his/her wish $w$ using a public hash function to generate an encryption key $K$. Using this key, the user encrypts $w$ and, separately, identity information and a session key for future communication, and submits the two ciphertexts to the public bulletin board for posting. Any user can download any pair of posted ciphertexts and verify whether his/her wishes match the posted ones and, if so, can obtain the session key for communicating with the matching partner. Although this simple protocol satisfies the first two requirements of Baldwin and Gramlich, it does not support joint notification of matches, since this property would require the (re)introduction of trusted third-parties [58].

The Zhang-Needham protocol faces two significant privacy challenges. First, an adversary can launch an *off-line dictionary attack* to discover the identity of a user who posted wishes on the public bulletin board. Because the wish space must be relatively small to allow straightforward user specification of wishes and clear-cut matches, an adversary can choose any set of possible wishes, hash them to produce an encryption key, and then decrypt the pairs of posted ciphertexts [58]. Thus, exhaustive use of all possible wishes is guaranteed to uncover a matching wish and the identities of the users posting it. Second, even if we make the (unrealistic) assumption that user wishes have large entropy, it is sufficient for an adversary to break the privacy of a posted wish to enable the compromise of *all* previous protocol executions containing that wish. In other words, this protocol does not provide *forward privacy* of users' identities and their wishes.

In this paper, we define the goals of *privacy-enhanced matchmaking* protocols by augmenting the original two requirements (i.e., anonymity of protocol users and authentication of wish matches[1]) with new security goals, which appear to be fundamental to private matchmaking. Our overall set of security goals comprises:

- authenticity of users and wish matches;
- privacy of users' identities and of their wishes; in particular:
    - anonymity of users and privacy of wish matches;
    - privacy resistance to off-line dictionary attacks; and
    - forward privacy of users' identities and their wishes.

These security goals are fundamental to privacy-enhanced matchmaking (and other similar) protocols. Authenticity is the basis for trust between users that matched their wishes as it prevents impersonation of legitimate users. Lack of wish privacy can lead to breaches of user privacy since wishes are typically specific to classes of users (e.g., known specific skill sets and other user characteristics, such as desired security clearances, can be linked with certain users and organizations). Resistance to off-line dictionary attacks is also fundamental because, in practice, wish entropy is fairly low: the space of user wishes is rather limited and fairly predictable thereby enabling potent off-line attacks. Finally, forward privacy is also important due to the *durability of privacy concerns*: a breach of privacy in a current protocol run should not cause the break of privacy of older runs (i.e., by analogy to the basic notion of perfect forward security of key exchange protocols).

In this paper, we also present a privacy-enhanced matchmaking protocol that provably counters any adversary that attempts to violate the privacy goals stated above. The protocol is based on a very simple construction that is efficiently implemented using a password-based authenticated key exchange (PAKE) protocol [10, 21, 35, 36, 40, 48, 50]. In addition, and as a side result of independent theoretical interest, we show that for any user authentication problem in which secrets are chosen from low entropy sets, two notions related to our last two goals, namely security against off-line dictionary attacks and forward security in the corruption model, are equivalent.[2]

## 2 Related Problems

In this section, we explore a variety of related problems and explain the differences between these problems and ours. In particular, we argue that solutions to these related problems are insufficient to solve our problem.

*Secret Handshakes.* The problem of secret handshakes is directly related to our problem. Secret handshakes allow two parties, which are suspicious about each other's affiliation, to securely recognize each other only if they have the same affiliation [3, 17, 52]. When compared to our problem, one can easily see secret handshakes is a specific instance of a privacy-enhanced matchmaking protocol (in other words, the latter is more general problem than the former). All secret handshake problems studied to date assume that each party uses classical cryptographic (i.e., high-entropy) keys that are distributed by a group manager prior to any execution of the protocol. In contrast, the use of low-entropy secrets (i.e., wishes) in our problem is an important practical requirement. Hence, in any secret-handshake setting where members in a same group are sharing a low-entropy password, our problem can provide a practical solution. Moreover, a secret handshake implemented from our solution can enjoy different flavors of communications as follows:

- with full anonymity: users communicate each other as long as they are convinced that they belong to a same group *without* ever being traced (i.e., identified), or
- with privacy-preserving entity authentication: once users are convinced they belong to the same group, they identify each other prior to further communication. or
- with traceability and anonymity: users can be traced by the group manager while full anonymity is preserved

[1] As pointed out by Zhang and Needham [58], support of joint notification of users only in the event of a positive match requires an on-line trusted authority, which we also want to avoid.

[2] Since it is already known that forward security in the corruption model is stronger than security against off-line dictionary attacks in the non-corruption model, we only need to show that security definition of password-based authenticated key exchange in the non-corruption model implies forward security in the corruption model (viz., Appendix B).

among users.[3]

However, existing solutions of secret handshakes do not fit into our problem as we cannot assume the use of high-entropy secrets, which is a fundamental requirement of all secret-handshake solutions.

*Set Intersection.* Set intersection allows two or more parties, each having a set of elements, to securely learn the intersection of their sets without revealing elements not in the intersection [41, 32]. The major difference between our problem and set intersection is that set intersection is not necessarily an *exact matching*. Therefore, by engaging in an interaction with an honest user on an input of a set including all possible elements, an adversary can determine the honest user's input with probability one. Therefore, set interaction cannot provide a secure solution for our problem.

*Trust-Negotiation.* Trust negotiation allows a client to access a server's resources without having to reveal all the client's credentials and disclose the complete server's access policy, provided by the server's policy is satisfied [8, 54, 51, 57, 56, 53]. Within an appropriate setting, our problem can be applied to each step of gradual negotiation to see whether each of a client's credentials exactly satisfies each access check of the server's policy. Furthermore, our solution can enhance client and server privacy so that their identities are not revealed until the last step of trust-negotiation is satisfied. Their identities are also kept anonymous to passive eavesdroppers. However, trust-negotiation solutions do not consider user (i.e., clients and servers) anonymity.

*Other Privacy-Preserving Problems.* Several problems have been introduced in the privacy-preserving area of access control. Hidden credentials [38, 5], oblivious envelopes [42, 49, 44, 45] and policy-based encryptions [6, 7] are relevant examples. However, they focus mainly on the privacy of entity's attributes, for example, affiliation, policy, etc., and do not consider all users' privacy concerns. In particular, all problems assume that users know each other's identity, while in our problem users' identities are not revealed unless they have a common (i.e., a matching) wish. For similar reasons, it is doubtful whether generic two-party secure computation protocols [55, 37, 33] can provide a solution of our problem; i.e., to date, all generic two-party secure computation protocols are carried out in settings where identities of two parties are known to each other (Nevertheless, the possibility of applying generic secure two-party computation protocols to solve anonymous communication problems represents an interesting open research problem).

## 3 Preliminaries and Assumptions

*Anonymous Communication Channels.* Like most other privacy-preserving protocols, privacy-enhanced matchmaking requires the use of anonymous communication channels. Use of ordinary communication channels is inadequate because anonymity and hence identity privacy (e.g., linking user actions) can be simply broken via eavesdropping on communication messages. Among other measures, anonymous communication relies on pseudonym-naming – a feature commonly provided by most privacy protocols. In practice, low-latency anonymous channels exist (e.g., Tor, JAP [29, 31]).

*Untrusted Matchmaker.* A matchmaker publishes description of matchmaking, roles and wishes. Also, the matchmaker binds a pseudonym to a user's address of anonymous communication channel and signs, distributes and revokes pseudonyms. However, the matchmaker is not trusted with the privacy of users. We assume that the matchmaker functions correctly.

*Protocol Users and Secret Wishes.* Let $\mathcal{U}$ be a fixed set of users who may participate in protocol executions. Although $\mathcal{U}$ is public information, we assume that users start communicating without knowing any information about each other's real identity but only know pseudonyms which are generated from a set of pseudonyms $\mathcal{I}$. Let $\mathcal{W}$ be a predefined set of publicly known wishes. We assume that a wish is a low-entropy secret and hence that $1/|\mathcal{W}|$ is small but non-negligible. For simplicity and clarity, we assume that a user chooses a wish uniformly at random. However, even when any arbitrary relation exists between wishes and users, and such relations are known to the adversary, security definitions can be adjusted appropriately as long as the following assumption holds: for each wish $w \in \mathcal{W}$, there exist at least two users $U_1$ and $U_2$ such that they are equally likely to use wish $w$ as an input.

## 4 Security of Privacy-Enhanced Matchmaking

We separate security requirements of our protocol into two classes namely those addressing on-line and off-line adversaries. Their goals and the means of countering them are different. While on-line attacks that try to break the protocol through "on-line" interaction have a non-negligible probability of success in discovering low-entropy secrets, their handling is provided by attack detection and prevention of further protocol executions (discussed in Section 4.1 below). In contrast, off-line adversaries are substantially more challenging since such adversaries' off-line attacks can neither be detected nor blocked. For example, off-line adversaries can launch *dictionary* attacks by trying all possible

[3] If we equip our protocol with a group signature scheme, we can implement a secret handshake protocol based on low-entropy passwords that fully satisfies the security properties of the extant notion of secret handshakes [3, 17, 52].

wishes from $\mathcal{W}$ on information that is obtained via passive eavesdropping. Hence, by separating the two types of adversary, we can focus primarily on handling the more potent *off-line adversaries*. [4]

*Definitions*. An honest user is allowed to execute an unlimited number of protocol instances. Further, a user has a unique pseudonym for each execution (i.e, for each session). Therefore, without loss of generality, we assume that a pseudonym assigned to each execution represents the instance of the execution. Although not specifically stated, an instance of a user execution of our protocol is always performed with a new user pseudonym (Also, we use the notions of sessions and instances interchangeably).

An input of the protocol consists of a user wish, a user pseudonym, (real) user identity, and a partner's pseudonym. For simplicity, we say a user $U$ *uses* a wish $w$ if $U$ takes $w$ as an input secret of the protocol. We say a user $A$ *accepts* a user $B$ if $A$ outputs $B$ at the end of the protocol execution, and it means that $A$ has recognized and authenticated $B$ as a matching-wish partner. We say users $A$ and $B$ *interact* when they are informed of each other's pseudonym and engage in a protocol execution.

We say that an adversary $\mathcal{A}$ is *given an interaction with an honest user U who is running the protocol on input* $w$, when (1) an instance of $U$ is initiated with inputs of wish $w$, its pseudonym $I$, its real identity $U$ and $\mathcal{A}$'s (i.e., partner's) pseudonym $I'$ and $I$ is known to $\mathcal{A}$; and further, (2) whenever upon receiving a message from $\mathcal{A}$, the next message of the instance is computed according to the protocol and sent to $\mathcal{A}$.

*Concrete Security Properties*. The security goals for private matchmaking are supported by several concrete security properties that counter both on-line and off-line adversary attacks. The concrete properties are summarized in Figure 1 and defined below.

### 4.1 Security Properties that Counter On-line Adversaries

An on-line adversary can use a private matchmaking protocol to detect the identity of a honest user by guessing correctly a user's wishes with small but non-negligible probability (e.g., the probability of a correct guess can be lowered, but only to a limited degree, by extending the size of the wish space). By requiring that the adversary present his/her non-anonymous credentials to an honest user *after* any wish match, the protocol ensures that the user can detect an unwarranted match (or an on-line attack); i.e., a match whereby the adversary cannot present valid identity and wish credentials. Upon detection of an on-line attack, the user can request the revocation of the adversary's (anonymous) credentials from the matchmaker. Using a signed transcript of the adversary-user interaction, the matchmaker requests revocation from the certification authority which issued the adversary's anonymous credentials. A valid user revocation request, would cause the matchmaker to deny issue of a valid (signed) pseudonym to the adversary since the adversary could no longer produce the necessary (anonymous) credentials to the matchmaker after revocation. Thus, further, on-line, anonymous wish guessing by an adversary is blocked. Of course, an honest user would not initiate the matchmaking protocol unless the adversary (or any honest user) produces a matchmaker-signed pseudonym.

Limiting an on-line adversary's protocol execution after an unwarranted wish match requires an initial user interaction with a trusted certification authority. We use an anonymous credential system (e.g., [13, 22]) so that the user (or adversary) can prove the validity of his/her credentials without revealing his/her identity to the matchmaker. This ensures that user privacy is protected with respect to the matchmaker. All the matchmaker knows is the identity of a certification authority it trusts. Upon receiving a valid anonymous credential from the user, the matchmaker produces a signed pseudonym for a single protocol execution. The matchmaker also keeps a log of the user's proof transcripts ($T$) along with the corresponding pseudonym ($p$).

Anonymous credentials are revoked as follows: if the matchmaker receives a report (i.e., signed transcript by a reporter and encrypted with CA's public key) that a user with a pseudonym $p$ guessed a wish but lacked appropriate credentials, the matchmaker finds the proof-transcript $T$ corresponding to the pseudonym $p$ and forwards the report along with $T$ to the certification authority with a signed request to revoke the credential of the user identified in $T$. The certification authority verifies the validity of the report (e.g., verifies the signatures) and revokes the user's credentials. Note that at no point of the revocation protocol does the matchmaker discover the identities of the user and adversary.

In the above revocation scenario, we did not distinguish between an adversary's non-anonymous identity and wish credentials required by an honest user upon a wish match. In the rest of this paper, we assume that the adversary is only required to produce a valid non-anonymous identity credential. An attack in which an adversary fails to produce such a credential to an honest user after a match would be significantly more likely than one in which the adversary produces a valid non-anonymous identity and invalid non-anonymous wish credentials. Nevertheless, we note that requiring verification of the adversary's (or any user's) non-anonymous wish credentials upon a wish match does not introduce any

[4] We choose to model our security requirements using game conditions [27] rather than realizing ideal functionality. Our choice is motivated by the fact: (1) sometimes, additional message steps are necessary to realize ideal functionality [16, 18, 21] and (2) sometimes, it is impossible to realize ideal functionality without extra set-up assumptions [16, 20, 27].

**Security Properties**

- *Impersonation resistance* counters on-line adversary attacks against user authentication.
- *Detector resistance* counters on-line adversary attacks against user and wish privacy, and match authentication (i.e., unless a match occurs, privacy of users' identities and their wishes are preserved).
- Security properties that counter off-line adversary attacks, namely *matching-result privacy, user unlinkability and wish unlinkability* provide privacy resistance to off-line dictionary attacks and forward privacy of users' identities and their wishes. In particular, *wish unlinkability* captures forward privacy of wishes and *user unlinkability* captures forward privacy of users' identities.

**Figure 1. Overview of Our Security Properties and Attacks Countered**

additional protocol interaction or complexity, and for this reason we ignore this case for the balance of the paper.

In the rest of this section, we define two security properties that counter on-line adversaries, namely impersonation resistance and detector resistance. Essentially, the former captures entity authenticity and the latter captures identity privacy.

*Impersonation Resistance.* Intuitively, impersonation resistance requires that an adversary who is not a legitimate user cannot authenticate itself as a legitimate user to any honest user. This property should hold no matter what secret wish the adversary uses in the impersonation attack (e.g., even when wishes are matching, the adversary should not be able to impersonate a legitimate user).

**Definition 1** Formally, we say a matchmaking protocol has *impersonation resistance* if, for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following experiment is negligible:

1. $\mathcal{A}$ selects a victim user $V$ and a target user $T$ from $\mathcal{U}$ and a wish $w$ from $\mathcal{W}$ ($\mathcal{A}$ will try to impersonate $V$ to $T$).[5]

2. Then, $\mathcal{A}$ is given an interaction with $T$ who is running the protocol on input $w$.

In the experiment, if $T$ accepts $V$ as a matching partner, we say $\mathcal{A}$ *wins*.

*Detector Resistance.* Intuitively, detector resistance captures the identity-privacy concern: given a single interaction of the adversary $\mathcal{A}$ with an honest user, $\mathcal{H}$, adversary $\mathcal{A}$ cannot learn the real identity of $\mathcal{H}$ unless $\mathcal{A}$ and $H$ execute the interaction on a same wish. We model this as an indistinguishability property.

**Definition 2** We say a matchmaking protocol has the *detector resistance* property if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following experiment is negligibly close to $\frac{1}{2} + \frac{1}{2|\mathcal{W}|}$:

1. A random coin $b$ is flipped. Two random users $U_0$ and $U_1$ are selected from $\mathcal{U}$ and a random wish $w$ is chosen from $\mathcal{W}$.

2. If $b = 0$, $\mathcal{A}$ is given an interaction with $U_0$ who is running the protocol on input $w$. If $b = 1$, $\mathcal{A}$ is given an interaction with $U_1$ who is running the protocol on wish $w$.

3. When the interaction is complete, $\mathcal{A}$ is given the real identities of users, ($U_0$, $U_1$) and $w$.

4. Finally, $\mathcal{A}$ outputs $b'$ (guessing whether $\mathcal{A}$ has an interaction with $U_0$ or $U_1$) and if $b' = b$, we say $\mathcal{A}$ *wins*.

## 4.2 Security Properties that Counter Off-line Adversaries

An off-line adversary is eavesdropping on honest executions and then trying off-line dictionary attacks on the obtained information. In this adversarial model, we introduce three relevant security properties, namely matching-result privacy, wish unlinkability, and user unlinkability.

*Matching-result Privacy.* Intuitively, when given a transcript of an honest execution between two users, the adversary cannot learn anything about the matching result of the execution; i.e., whether two users engaged in the execution on a common wish.

**Definition 3** We say a matchmaking protocol has *matching-result privacy* if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following experiment is negligibly close to $\frac{1}{2}$:

1. Two random users $U_1$ and $U_2$ are selected from $\mathcal{U}$. A coin bit $b$ is flipped.

    - If $b = 0$, a random wish $w$ is chosen. Then, an honest interaction between users $U_1$ and $U_2$, both

[5] We allow $\mathcal{A}$ to choose the wish $w$, because we want impersonation resistance property to hold even when the adversary impersonating $V$ uses a same wish that $T$ uses.

on input wish $w$, is executed and the execution transcript is given to $\mathcal{A}$.

- If $b = 1$, two random wishes $w_1$ and $w_2$ are chosen from $\mathcal{W}$. Then, an honest interaction between $U_1$ and $U_2$, on input wishes $w_1$ and $w_2$, respectively, is executed, and the execution transcript is given to $\mathcal{A}$.[6]

2. Finally, $\mathcal{A}$ outputs a bit $b'$ and if $b' = b$, we say $\mathcal{A}$ *wins*.

*Wish Unlinkability.* Wish unlinkability captures forward privacy of wishes. Intuitively, wish unlinkability requires that the adversary cannot tell in which executions $w$ has been used as an input wish.

**Definition 4** We say a matchmaking protocol has *wish unlinkability* if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following experiment is negligibly close to $\frac{1}{2}$:

1. Two different wishes $w$ and $w'$ are randomly selected from $\mathcal{W}$ and given to $\mathcal{A}$. Four users $U_0, U_1, U_2$ and $U_3$ are chosen from $\mathcal{U}$. A coin bit $b$ is flipped.

   - If $b = 0$, an honest interaction between users $U_0$ and $U_1$ on input wish $w$, and another honest interaction between $U_2$ and $U_3$ on input wish $w$ are executed and the execution transcripts are given to $\mathcal{A}$.
   - If $b = 1$, an honest interaction between $U_0$ and $U_1$ on input wish $w$ and another honest interaction between $U_2$ and $U_3$ on input wish $w'$ are executed, and the execution transcripts are given to $\mathcal{A}$.

2. Finally, $\mathcal{A}$ outputs a bit $b'$, and if $b' = b$, we say $\mathcal{A}$ wins.

*User Unlinkability.* User unlinkability captures forward privacy of users' identities. Intuitively, user unlinkability requires that, when given a transcript of an execution run by a particular user whose real identity is $U$, the adversary cannot detect whether a new execution transcript belongs to the user $U$. It should hold even though the adversary has learned wishes used in the executions.

**Definition 5** We say a matchmaking protocol has *user unlinkability* if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following experiment is negligibly close to $\frac{1}{2}$:

1. Four different users $U, U_0, U_1, U_2$ are randomly selected from $\mathcal{U}$ and two wishes $w, w'$ are randomly selected from $\mathcal{W}$. $U$, $w$ and $w'$ are given to $\mathcal{A}$. A coin bit $b$ is flipped.

   - If $b = 0$, an honest interactions between users $U$ and $U_0$ on input wish $w$, and another honest interaction between $U$ and $U_1$ on input wish $w'$ are executed and the execution transcripts are given to $\mathcal{A}$.
   - If $b = 1$, an honest interaction between $U$ and $U_0$ on input wish $w$, and another honest interaction between $U_2$ and $U_1$ on input wish $w'$ are executed and the execution transcripts are given to $\mathcal{A}$.

2. Finally, $\mathcal{A}$ outputs a bit $b'$, and if $b' = b$, we say $\mathcal{A}$ wins.

Given all the security properties, we define a privacy-enhanced matchmaking protocol.

**Definition 6** We say a matchmaking protocol is a *privacy-enhanced* if the protocol has impersonation resistance, detector resistance, matching-result privacy, wish unlinkability and user unlinkability.

# 5 Protocol Design

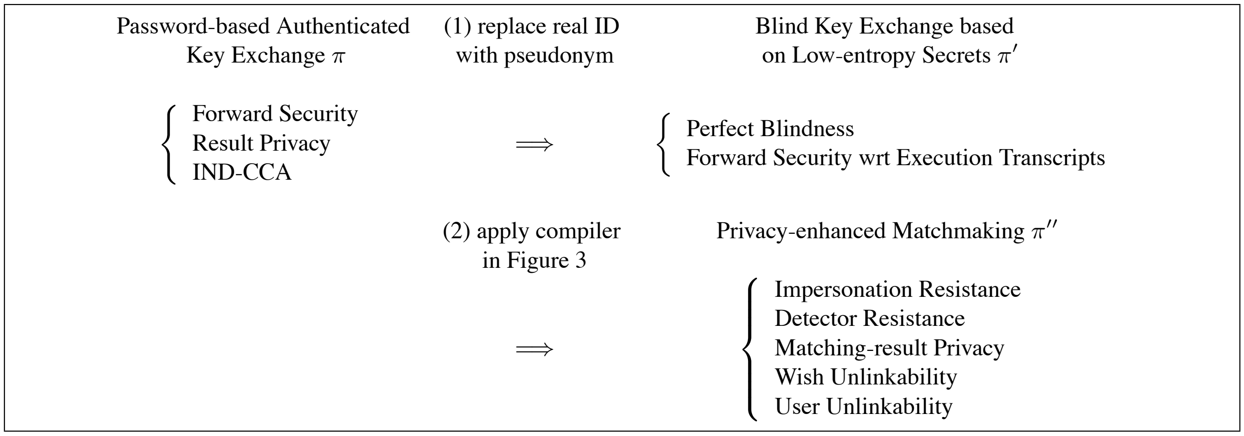
We design a privacy-enhanced matchmaking protocol in a multi-step modular way. First, we take a password-based authenticated key exchange (PAKE) protocol $\pi$ satisfying certain properties that are useful in building our solution (We briefly summarize the notion of PAKE and the efficiency of existing solutions in Appendix A). Then, we generalize passwords into low-entropy secrets (i.e., wishes) and add *perfect blindness* by simply replacing user identity field with pseudonym. It will result in a protocol named "blind key exchange based on low-entropy secrets" or BKE-LS in short. Finally, we transform a BKE-LS to a privacy-enhanced matchmaking protocol by adding back entity authentication (which was removed by adding perfect blindness) in a way of providing entity privacy (i.e., confidentiality). For an overview, our procedure to obtain a solution is illustrated in Figure 2. We describe each step in detail in the following sections. Note that we omit the revocation protocol for on-line adversaries and assume that such an adversary is limited to a single unwarranted wish match (viz., Section 4 above).

## 5.1 Relevant PAKE Security Properties

The PAKE security properties relevant to our protocol are forward security, result privacy, and tight IND-CCA of session key encryption.[7] For these properties, we only focus on *off-line dictionary attackers* which are given transcripts of executions between honest players. We show that these

[6] A stronger notion of matching-result privacy is possible by letting the adversary know secret wishes and it is achievable by our construction. However, the current notion is sufficient for our purposes.

[7] We assume that the reader is familiar with the definition of secure PAKE protocols and related notation. For details, we refer the reader to references [10, 40].



Password-based Authenticated Key Exchange $\pi$

Forward Security<br>
Result Privacy<br>
IND-CCA

(1) replace real ID with pseudonym

$\Longrightarrow$

Blind Key Exchange based on Low-entropy Secrets $\pi'$

Perfect Blindness<br>
Forward Security wrt Execution Transcripts

(2) apply compiler in Figure 3

$\Longrightarrow$

Privacy-enhanced Matchmaking $\pi''$

Impersonation Resistance<br>
Detector Resistance<br>
Matching-result Privacy<br>
Wish Unlinkability<br>
User Unlinkability

**Figure 2. Transformations to obtain Privacy-enhanced Matchmaking $\pi''$ from PAKE $\pi$**

properties provided by a password-based key exchange protocol (PAKE) and hence a PAKE proven secure in the non-corruption model is sufficient to our solution.

*Forward Security* [10, 40]. Intuitively, forward security implies that corruption of a user's password does not break the security of sessions keys used prior to the corruption. This notion has already been introduced in the authenticated key exchange problem where a long-standing belief has been that forward security in the weak corruption model (where the adversary is allowed to corrupt a user's long-term key, or password)[8] is strictly stronger than security in the non-corruption model (where corruption of long-term key, or password, is not allowed). However, in a password-only (i.e., low-entropy secret) setting, we show that any PAKE protocol secure in the non-corruption model also has forward security in the weak corruption model.

**Theorem 1** *PAKE security in the non-corruption model implies forward security in the weak corruption model.*

The proof of this theorem is provided in Appendix B.

*Result Privacy.* Intuitively, result privacy captures the following property: when the adversary passively observes an interaction between two honest users where the adversary does not know whether the users' passwords are equal, the adversary should not be able to tell whether the two honest users have accepted the same session key. If the adversary can learn that the two honest users have *not* accepted the same session key, then the adversary knows that the users' passwords are different. This notion of result privacy has not received attention before, because for the authentication problem, (1) it is natural for two parties to share the same password (e.g., in advance, by registration), to interact with each other, and (2) two parties who have already had a successful interaction are likely to have further communication, so the success of the result matching would become known to the adversary, anyway. In contrast, the notion of matching-result privacy is an important security property of our problem. Also, in further contrast with traditional PAKE applications, subsequent communication between two users who had a successful match of wishes is also supposed to be anonymous. Therefore, in our problem the result of an interaction between two users cannot possibly become trivially learnable information by an adversary. Hence, result privacy is a relevant property for a PAKE protocol whenever that protocol is used as a building block for privacy-enhanced matchmaking.

**Definition 7** We say a protocol has *result privacy* if, for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following game is negligibly close to $\frac{1}{2}$: a coin $b$ is flipped; if $b = 0$, a transcript of an honest execution between two random users such that their passwords are different is given to $\mathcal{A}$. If $b = 1$, a transcript of an honest execution between two random users such that their passwords are same is given to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a guess bit $b'$ and wins if $b' = b$.

PAKE protocols *without* explicit authentication (i.e., with only implicit authentication) satisfy the result privacy as shown by the following theorem.

**Theorem 2** *Any PAKE protocol with implicit authentication satisfies result privacy.* [9]

[8] Here, we only consider the *weak* corruption which, in contrast with the *strong* corruption model, does not allow the adversary to have a complete control over users.

[9] So far, our result privacy notion only considered security against passive eavesdroppers explicitly. However, result privacy against active adversaries (i.e., impersonators) is clearly satisfied by the definition of *on-line adversaries*. Intuitively, the definition of on-line attack captures that once the adversary carried out an on-line attack against an honest user by guessing a password and engaging in an execution with the user, the adversary cannot tell whether the guess was correct until the adversary corrupts either

The proof of this theorem is similar to that of Theorem 1 (except for some technical details) and hence is omitted.

*Tight IND-CCA of Session Key-based Encryption.* It is well-known (e.g., [10]) that a common session key established between two parties via authenticated key exchange, allows them to have a secure future communication enhanced with either authenticity or confidentiality (or both). For example, by applying the common session key to a symmetric key encryption scheme that has indistinguishability against chosen ciphertext attack (IND-CCA), two parties can communicate each other without losing confidentiality.

Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ denote any IND-CCA symmetric key encryption scheme. Informally, *tight IND-CCA* with respect to $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ implies that no PPT adversary can distinguish a ciphertext of $m_0$ from a ciphertext of $m_1$, where the ciphertexts are encrypted with a session key $\mathsf{sk}$ (i.e., $\mathcal{E}_{\mathsf{sk}}(m_b)$) and messages $m_0$ and $m_1$ are chosen by the adversary. In particular, we want the highest probability that the adversary breaks this property to be negligibly close to the probability that the adversary distinguishes a real session key from a random key. In PAKE protocol definitions [40, 10], this corresponds to the event that the adversary *succeeds* with probability negligibly close to $\frac{1}{2} + \frac{1}{2N}$, where $N$ is the size of low-entropy secret set, whenever the protocol is a secure.[10] More formally, we define a new experiment where adversary $\mathcal{A}$ is given all the oracles except the $\mathsf{Test}$ oracle ; viz., the experiment of the PAKE security definition [10, 40].[11] Additionally, we define a new oracle $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}$ as follows:

- $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}(m_0, m_1, \Pi_U^i)$: Upon receiving two messages $m_0$ and $m_1$ and an instance $\Pi_U^i$ from $\mathcal{A}$, a bit $b$ is flipped and $\mathcal{E}_{\mathsf{sk}_i}(m_b)$ is given to $\mathcal{A}$ where $\mathsf{sk}_i$ is the session key of $\Pi_U^i$.

Finally, in the experiment, $\mathcal{A}$ outputs a bit $b'$ and wins if $b' = b$.

**Definition 8** We say a protocol $\pi$ has *tight IND-CCA with respect to* an encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the game of $\pi$ with the $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}$ and given encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is negligibly close to $\frac{1}{2} + \frac{1}{2N}$, where $N$ is the size of low-entropy secret set.

Authenticated key exchanges based on high-entropy secrets (e.g., symmetric-key based key exchange and public-key based key exchange) easily imply the *tight IND-CCA* property, because the probability that $\mathcal{A}$ breaks the protocol is negligible. However, this is *not* trivially true in the case of password-based key exchange where passwords are *low-entropy secrets* because the probability that $\mathcal{A}$ breaks the protocol is non-negligibly high. However, it can be shown that most of existing solutions satisfy the tight IND-CCA property in a *non-black box* way.

## 5.2 Generalizing Passwords as Low-entropy Secrets and Adding Perfect Blindness

In this section, we modify a password-based authenticated key exchange to obtain a blind key exchange based on low-entropy secrets (BKE-LS). Our main task is providing a perfect blindness by breaking the binding between secrets (e.g., passwords, wishes) and user IDs. Blind key exchange based on low-entropy secrets is obtained by adding perfect blindness to a password-based authenticated key exchange so that it will provide *no* entity authentication. Here, we focus on providing *anonymity*; however, we do add back entity authentication as the final step of our protocol design (viz., the next section).

In the password-based authenticated key exchange protocol, authentication is provided upon the assumption that there is a binding between *user and user ID* (a user has a unique ID value), and a binding between *user and password* (each user has one password). (Typically these bindings are the result of the user registration process.) Therefore, a user identified by an user ID can be authenticated by password verification. In contrast, in our problem the low entropy secret, namely the "wish", is not used for user authentication. In particular, a user's wish is not necessarily fixed or registered in advance. Here, we generalize passwords as low-entropy secrets and we call them "wishes". We allow a user to use a different secret (i.e., a wish) for each execution and remove the restriction that the low-entropy secret has to be initialized prior to protocol execution (as in the case of passwords).

Finally, to provide *perfect blindness*, we remove the binding between user IDs and secret wishes by breaking the connection between user and user ID. In particular, we let a user have a new pseudonym instead of its (real) user ID for the ID field in each execution of the protocol. Hence the user ID field does not reveal anything about either the user or the secret (i.e., wish) used.[12]

To provide wish unlinkability in our solution, we introduce the notion of forward security with respect to execution transcripts. Intuitively, for any particular secret wish $w$, the adversary should not be able to tell whether an execution transcript has resulted from input $w$.

**Definition 9** We say a key exchange protocol has *forward security with respect to transcripts* if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following game is negligibly close to $\frac{1}{2}$: (1) Two different secrets $s_0, s_1$ are

the user or the session key that the user has accepted in protocol execution [40, 10].

[10] This is a standard result of secure PAKE protocols; viz., [10, 40].

[11] See the summary of the oracle definitions in Appendix B.

[12] In fact, the real user ID is never used in an execution of the BLK-LS protocol. However, the real user ID will be added in a privacy-preserving way later in the last step of our protocol.

randomly selected from $\mathcal{W}$ and given to $\mathcal{A}$. (2) A coin bit $b$ is flipped and two users $U_1$ and $U_2$ are selected. If $b = 0$, an execution between $U_1$ and $U_2$ on $s_0$ is simulated and the transcript is given to $\mathcal{A}$. If $b = 1$, an execution between $U_1$ and $U_2$ on input secret $s_1$ is simulated and the transcript is given to $\mathcal{A}$. (3) Finally, $\mathcal{A}$ outputs $b'$ and we say $\mathcal{A}$ wins if $b' = b$.

**Theorem 3** *If a protocol $\pi$ is a secure PAKE protocol, a BKE-LS protocol $\pi'$ obtained from $\pi$ has forward security with respect to transcripts.*

The proof of this theorem is similar to the of Theorem 1 and hence is omitted.

## 5.3 Final Step of Building a Privacy-enhanced Matchmaking Protocol

The compiler transforming a PAKE protocol $\pi$ into a BKE-LE protocol $\pi'$ and then into a PMM protocol $\pi''$ is illustrated in Figure 3. In this section, we briefly describe the last transformation from a BKE-LE protocol to a privacy-enhanced matchmaking protocol. The compiler essentially adds secure authentication between two parties, $U$ and $U'$. User $U$ runs BKE-LS $\pi'$ until it computes a session key $\mathsf{sk}$. Then, $U$ computes a digital signature $\sigma$ on transcripts of the execution of $\pi'$ (i.e., ordered concatenation of all the messages sent and received during the execution) and its real identity information including its real identity, its public key and the certificate of the public key. Further, user $U$ encrypts all the transcript, its real identity and the signature $\sigma$ with key $\mathsf{sk}$, and sends the ciphertext to party $U'$ with whom $U$ interacted during the execution of $\pi$. Upon receiving a ciphertext from $U'$, $U$ decrypts it with key $\mathsf{sk}$ and, if the plaintext is valid, $U$ verifies that (1) the decrypted transcript is the same as the original, and (2) the digital signature is valid using public key of $U'$. If the verification is all correct, $U$ accepts $U'$ as a matching partner. Otherwise, $U$ accepts no one.

**Theorem 4** *If $\pi$ is a secure PAKE protocol, then protocol $\pi''$ obtained by applying the compiler of Figure 3 to $\pi$ is a secure privacy-enhanced matchmaking protocol.*

**Proof** We give a sketch of the proof that each security property of privacy-enhanced matchmaking protocol is satisfied.

*Impersonation Resistance.* If there exists an adversary $\mathcal{A}$ that can break the impersonation resistance property of $\pi''$ with non-negligible probability $\delta$, then we can easily construct an algorithm $\mathcal{F}$ that breaks the underlying signature scheme $\Sigma$ with a probability at least $\delta$. Basically, $\mathcal{F}$ simulates a view for $\mathcal{A}$ and outputs a forged signature $\sigma'$, whenever $\mathcal{A}$, impersonating $V$ to an honest player $T$, outputs a forged, but valid signature $\sigma'$ for a uncorrupted user $V$. Then, the probability:

$$\Pr[\mathcal{F} \text{ forges a valid signature } \sigma' \text{ with respect to } V\text{'s public key}]$$

is at least $\Pr[T \text{ accepts } V]$, which is equal to $\delta(k)$. Since we assumed that $\delta(k)$ is non-negligible, it contradicts the assumption of security of the underlying digital signature scheme $\Sigma$.

*Detector Resistance.* If there exists an adversary $\mathcal{A}$ that can break the detector resistance property of $\pi''$ (*nb.* in a single interaction) with probability $\frac{1}{2} + \frac{1}{2|\mathcal{W}|} + \delta$ for a non-negligible function $\delta(k)$, then we can construct an algorithm $\mathcal{B}$ that breaks the *tight IND-CCA* property of $\pi$. $\mathcal{B}$ is given $\mathsf{Execute}$, $\mathsf{Send}$, $\mathsf{Reveal}$ and $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}$ and proceeds as follows:

1. $\mathcal{B}$ uses its own oracle $\mathsf{Send}$ to initiate an instance $\Pi'$ for a new random identity $I'$ and simulates $\mathcal{A}$'s view until the instance outputs a session key $\mathsf{sk}$ in the execution.

2. To simulate the last outgoing message (of $\pi''$), $\mathcal{B}$ carries out the following actions:

   (a) Obtain the secret $pw'$ of $I'$ by calling $\mathsf{Corrupt}(I')$.

   (b) Choose two different users $U_0$ and $U_1$ from $\mathcal{U}$ at random.

   (c) For each case of $b = 0$ and $b = 1$:

      i. Compute a signature $\sigma_b$ by signing a message $\mathcal{T}||U_b$, where $\mathcal{T}$ is the transcript of $\Pi'$.

      ii. Compose a message $m_b = \mathcal{T}||U_b||\sigma_b||\mathsf{info}_b$ where $\mathsf{info}_b$ denotes user $U_b$'s information.

   (d) Then, obtain a challenge ciphertext $\mathcal{C}$ by calling $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}(m_0, m_1, \Pi')$ and finish the interaction by sending the last message $\mathcal{C}$ to $\mathcal{A}$.

3. Finally, $\mathcal{B}$ gives $(U_0, U_1)$ and $pw'$ to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ outputs.

The simulation by $\mathcal{B}$ is perfect from $\mathcal{A}$'s perspective for the following two reasons. First, the parts of $\pi'$ are simulated by asking queries to $\mathsf{Send}$ oracle. Second, for the part of $\pi''$ (i.e., producing the ciphtertext $\mathcal{C}$), $\mathcal{B}$ itself learns the secret of $I'$ via $\mathsf{Corrupt}$ oracle query and so $\mathcal{B}$ computes a correct form of plaintext message (which is perfect since $U_0$ and $U_1$ are totally independent from $I'$, the pseudonym used in $\pi'$) and obtains a correct form of ciphertext $\mathcal{C}$ via the $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}(m_0, m_1, \Pi')$ query. Also, since $\mathcal{B}$ queries $\mathsf{Corrupt}(I')$ only after $\mathcal{B}$ finishes queries to the $\mathsf{Send}$ oracle, $\Pi'$ (i.e., instance of $I'$) is fresh. Moreover, by the definition of on-line attacks [10, 40], $\mathcal{B}$ makes only one on-line attack.

**Compiler**

Let $k$ be a security parameter. Let $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme which is existentially unforgeable against adaptive chosen-message attack. Let $\{PK_{U_i}, SK_{U_i}\}_{U_i \in \mathcal{U}}$ be a list of public/secret key pairs generated from $\mathsf{Gen}(1^k)$, and assume $\mathcal{U}$, $\{PK_{U_i}\}_{U_i \in \mathcal{U}}$ is publicly-known. Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a symmetric-key encryption scheme that is IND-CCA secure.

**The Protocol $\pi'$:** In $\pi$, let $ID_U$ be a variable indicating the identity of user $U$ and $pw_U$ be a variable indicating the password of $U$, and $\mathsf{pid}$ be a variable indicating partner ID. Given the input values, namely a wish $w$, a pseudonym $I$ and a partner's pseudonym $pI$, user $U$ verifies if the partner's pseudonym $pI$ is valid (signed by the matchmaker). If it is valid, user $U$ sets $ID_U = I, pw_U = w$, and $\mathsf{pid} = pI$ and runs protocol $\pi$ on those inputs.

**The Protocol $\pi''$:** In $\pi'$, if $U$ terminates accepting a session key $\mathsf{sk}$, $U$ keeps $\mathsf{sk}$. Otherwise, $U$ obtains a random key $r$ through $\mathcal{G}(1^k)$, and sets $\mathsf{sk} = r$. Given $\mathsf{sk}$, $U$ performs the following additional steps:

1. Let $\mathcal{T}$ be a concatenation of messages that $U$ has sent and received during the execution of $\pi'$. $U$ computes:
   (a) a signature $\sigma$ by signing a message $\mathcal{T}||U$, where $||$ denotes a concatenation of messages (i.e., $\mathsf{Sign}_{SK_U}(\mathcal{T}||U)$).
   (b) a ciphertext $C$ by encrypting a plaintext $M = \mathcal{T}||U||\sigma||\mathsf{info}$ with $\mathsf{sk}$ (i.e., $\mathcal{E}_{\mathsf{sk}}(M)$), where $\mathsf{info}$ is U's information that includes U's public key and the certificate of the public key.
2. $U$ sends ciphertext $C$ to a partner whose pseudonym is $pI$.
3. Upon receiving a ciphertext $C'$ from partner $pI$, $U$ decrypts it with $\mathsf{sk}$, obtains $\mathcal{T}'||U'||\sigma'||\mathsf{info}'$, and proceeds as follows:
   (a) If there is no public key for $U'$ or $\mathcal{T}' \neq \mathcal{T}$, $U$ terminates with a private output $\perp$. Otherwise, $U$ verifies $\sigma'$ by computing $\mathsf{Vrfy}_{PK_{U'}}(\mathcal{T}'||U', \sigma)$.
   (b) If the signature is not valid, $U$ terminates with a private output $\perp$. Otherwise $U$ terminates with a private output $U'$ (i.e., $U$ accepts $U'$ as a matching partner).

**Figure 3. Compiler to be applied to PAKE protocol $\pi$ to yield privacy-enhanced matchmaking protocol $\pi''$.**

For the analysis, let $\mathsf{Enc}_0$ denote the case that $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}$ oracle returns encryption of $m_0$ and $\mathsf{Enc}_1$ denote the case that $\mathsf{Test}_{\mathsf{IND\text{-}CCA}}$ oracle returns encryption of $m_1$. Then, since we have

$$\begin{aligned} \Pr[\mathcal{B} = 0|\mathsf{Enc}_0] &= \Pr[\mathcal{A} = 0|\mathsf{Enc}_0], \text{and} && (1) \\ \Pr[\mathcal{B} = 1|\mathsf{Enc}_1] &= \Pr[\mathcal{A} = 1|\mathsf{Enc}_1], && (2) \end{aligned}$$

the probability that $\mathcal{B}$ wins equals the probability that $\mathcal{A}$ wins, which is non-negligibly higher than $\frac{1}{2} + \frac{1}{2|\mathcal{W}|}$ (by the assumption), and it contradicts the fact that $\pi$ has the *tight IND-CCA* property with respect to $(\mathcal{G}, \mathcal{E}, \mathcal{D})$. (If $\pi$ has the *tight IND-CCA* property, an on-line attack can be successful only with a probability negligibly close to $\frac{1}{2} + \frac{1}{2|\mathcal{W}|}$).

*Security against off-line Adversaries.* Matching-result privacy, wish unlinkability and user unlinkability are clearly satisfied by security properties of the underlying PAKE (and so BKE-LS) protocol. In particular, matching-result privacy is guaranteed by result privacy of $\pi$. Wish unlinkability is preserved due to forward security with respect to execution transcripts and perfect blindness of $\pi'$. Finally, user unlinkability is obtained by perfect blindness of $\pi'$, forward security and tight IND-CCA property with respect to $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ that $\pi$ has. ■

## 6 Acknowledgements

This research was supported in part by US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001 and by the US Army Research Office under Contract W911NF-07-1-0287 at the University of Maryland. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, US Army Research Office, the U.S. Government, the UK Ministry of Defense, or the UK Government.

## References

[1] M. Boyarsky. Public-Key Cryptography and Password Protocols: The Multi-User Case. *ACM CCS*, 1999.

[2] M. Bellare, R. Canetti, and H. Krawczyk. A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols. *STOC* '98.

[3] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H. Wong. Secret Handshakes from pairing-based key agreement. *IEEE Security and Privacy*, 2003.

[4] R.W. Baldwin and W.C. Gramlich, Cryptographic Protocol for Trustable Match Making, *IEEE Security and Privacy*, 1985.

[5] R. Bradshaw, J. Holt, and K. Seamons. Concealing complex policies with hidden credentials. In *Proceedings of 11th ACM Conference on Computer and Communications Security*, Oct. 2004.

[6] W. Bagga and R. Molva. Policy-Based Cryptography and Applications. In *9th International Conference on Financial Cryptography and Data Security*, 2005.

[7] W. Bagga, R. Molva and S. Crosta. Policy-Based Encryption Schemes from Bilinear Pairings. In *ACM Symposium on Information, Computer and Communications Security*, 2006.

[8] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pp 134-143. 2000.

[9] V. Boyko, P. MacKenzie, and S. Patel. Provably Secure Password Authentication and Key Exchange Using Diffie-Hellman. *Advances in Cryptology - Eurocrypt*, 2000.

[10] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure Against Dictionary Attacks. *Eurocrypt* 2000.

[11] M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. *Crypto* '93. Springer-Verlag, 1993, pp. 232–249.

[12] M. Bellare and P. Rogaway. Provably-Secure Session Key Distribution: the Three Party Case. *STOC* '95.

[13] D. Chaum. Security without Identification: Transaction Systems to make Big Brother Obsolete. *Communications of the ACM*, 1985.

[14] L. Chen. Access with Pseudonyms. *Cryptography: Policy and Algorithms*, 1995.

[15] D. Chaum and J. Evertse. A Secure and Privacy-Protecting Protocol for Transmitting Personal Information between Organizations. *Crypto* 1986.

[16] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Manuscript dated Jan. 28, 2005, available at http://eprint.iacr.org/2000/067. A preliminary version appeared in *FOCS* 2001.

[17] C. Castelluccia, S. Jarecki, G. Tsudik. Secret Handshakes from CA-Oblivious Encryption, *Asiacrypt*, 2004.

[18] R. Canetti and H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. *Eurocrypt* 2001.

[19] R. Canetti and H. Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. *Eurocrypt* 2002. Full version available at http://eprint.iacr.org/2002/059.

[20] R. Canetti, E. Kushilevitz, and Y. Lindell. On the Limitations of Universally Composable Two-Party Computation Without Set-up Assumptions. *Eurocrypt* 2003.

[21] R. Canetti, S. Halevi, J. Katz, Y. Lindell, and P. MacKenzie, Universally Composable Password-Based Key Exchange, *Eurocrypt*, 2005.

[22] J. Camenisch and A. Lysyanskaya. Efficient Non-transferable Anonymous Multi-show Credential System with Optional Anonymity Revocation. *Eurocrypt*, 2001.

[23] J. Camenisch and A. Lysyanskaya. A Signature Scheme with Efficient Protocols. *Security in Communication Networks*, 2002.

[24] J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. *Crypto* 2004.

[25] R. Cramer and V. Shoup. Universal Hash Proof and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. *Eurocrypt*, 2002.

[26] I. B. Damgard. Payment Systems and Credential Mechanism with Provable Security against Abuse by Individuals. *Crypto* 1988.

[27] A. Datta, A. Derek, J. Mitchell, A. Ramanathan, A. Scedrov. Games and the Impossibility of Realizable Ideal Functionality. *Theory of Cryptography Conference (TCC)* 2006.

[28] G. Danezis, R. Dingledine and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. *IEEE Symposium on Security and Privacy, Berkeley, CA*, 2003.

[29] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. *Proceedings of the 13th USENIX Security Symposium*, 2004.

[30] W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Trans. Info. Theory* 22(6): 644–654 (1976).

[31] H. Federrath, S. Kopsell and R. Wendolsky. Project: AN.ON - Anonymity.On-line, JAP. *http://anon.inf.tu-dresden.de/*

[32] M. Freedman, K. Nissim, B. Pinkas, Efficient Private Matching and Set Intersection, *Eurocrypt*, 2004.

[33] O. Goldreich. Foundations of Cryptography: Volume 2 - Basic Applications. Cambridge University Press, 2004.

[34] R. Gennaro. Faster and Shorter Password-Authenticated Key Exchange. http://eprint.iacr.org/2007/325, 2007.

[35] R. Gennaro and Y. Lindell. A Framework for Password-Based Authenticated Key Exchange. *Eurocrypt* 2003.

[36] O. Goldreich and Y. Lindell. Session-Key Generation using Human Passwords Only. *Crypto*, 2001.

[37] O. Goldreich, S. Micali, and A. Wigderson. How to Play any Mental Game, or A Completeness Theorem for Protocols with Honest Majority. *STOC* 1987.

[38] J. E. Holt, R. W. Bradshaw, K. E. Seamons, and H. Orman. Hidden credentials. In *Proceedings of the 2nd ACM Workshop on Privacy in the Electronic Society,* Oct. 2003.

[39] S. Halevi and H. Krawczyk. Public-Key Cryptography and Password Protocols. *ACM Trans. on Information and Systems Security*, 2(3):230-268, 1999.

[40] J. Katz, R. Ostrovsky, and M. Yung. Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. *Eurocrypt*, 2001.

[41] L. Kissner, D. Song. Private and Threshold Set-Intersection. CMU TR, 2004.

[42] N. Li, W. Du, and D. Boneh. Oblivious signature-based envelope. In *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing (PODC 2003)*, pp.182-189, Boston, Massachusetts, July 2003. ACM Press.

[43] T.M.A. Lomas, L. Gong, J.H. Salzer, and R.M. Needham. Reducing Risks from Poorly-Chosen Keys. *ACM Operating Systems Review* 23(5): 14-18, 1989.

[44] J. Li and N. Li. OACerts: Oblivious Attribute Certificates. In *Proceedings of the 3rd Conference on Applied Cryptography and Network Security (ACNS)*, vol 3531 of LNCS, 2005.

[45] J. Li and N. Li. A Construction for General and Efficient Oblivious Commitment Based Envelope Protocols. In *Proceedings of 8th International Conference on Information and Communications Security (ICICS)*, pp. 122-138. 2006.

[46] A. Lysyanskaya, R. Rivest, A. Sahai and S. Wolf. Pseudonym Systems. *Selected Areas in Cryptography*, 1999.

[47] C. Meadows, A More Efficient Cryptographic Matchmaking Protocol for Use in the Absence of a Continuously Available Third Party, *IEEE Security and Privacy*, 1986.

[48] P. MacKenzie, S. Patel, and R. Swaminathan. Password-Authenticated Key Exchange Based on RSA. *Asiacrypt*, 2000.

[49] S. Nasserian, G. Tsudik. Revisiting Oblivious Signature-Based Envelopes. Available at http://eprint.iacr.org/2005/283.

[50] M.H. Nguyen and S. Vadhan. Simpler Session-Key Generation from Short Random Passwords. *TCC*, 2004.

[51] K. E. Seamons, M. Winslett, and T. Yu. Limiting the disclosure of access control policies during automated trust negotiation. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS01)*, 2001.

[52] G. Tsudik and S. Xu, A Flexible Framework for Secret Handshakes, *PET(Privacy Enhancing Technologies)*, 2006 (earlier version was appeared in PODC 2005).

[53] W.H. Winsborough and N.Li. Safety in automated trust negotiation. In *Proceedings of the IEEE Symposium on Security and Privacy*, pp.147-160. 2004.

[54] W. H. Winsborough, K. E. Seamons, and V.E. Jones. Automated trust negotiation. In *DARPA Information Survivability Conference and Exposition*, vol 1, pp.88-102, IEEE Press, 2000.

[55] A.C.-C. Yao. How to Generate and Exchange secrets. *FOCS* 1986.

[56] T. Yu and M. Winslett. Unified scheme for resource protection in automated trust negotiation. In *Proceedings of IEEE Symposium on Security and Privacy*, pp. 110-122. IEEE Computer Society Press, 2003.

[57] T. Yu, M. Winslett, and K. E. Seamons. Interoperable Strategies in Automated Trust Negotiation. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pp.146-155. AMC Press, 2001.

[58] K. Zhang and R. Needham. A Private Matchmaking Protocol. http://citeseer.nj.nec.com/71955.html, 2001.

# A Password-based authenticated key exchange (PAKE)

Password-based authenticated key exchange allows two parties holding only short, human-memorable passwords to establish a secure session key of high-entropy when they share the same password. Such a key exchange is authenticated in a sense that it is secure against man-in-the-middle adversaries. While on-line attackers can guess a password with non-negligible probability, prevention of on-line attackers is straightforward with other mechanism (e.g., access block after consecutive log-in failures), it is not easy to prevent off-line attacker from enumerating all possible passwords of small space into execution transcripts. Therefore, essentially, major security property of password-based authenticated key exchange is security against off-line dictionary attackers.

## A.1 Efficiency of PAKE

Although very efficient PAKE constructions exist [10, 9], they rely on the idealized assumptions such as the ideal cipher and random oracle model. Those solutions only provide heuristic security when the random oracle is replaced by a public function such as SHA-1.

The KOY protocol by Katz *et al.* [40] and the GL protocol by Gennaro and Lindell — a generalization of the KOY protocol [35] — are PAKE constructions in the common reference string model and do not require any idealized assumptions.

According to the efficiency analyze in [40], each user only needs roughly 7-8 exponentiation computations. The cost is around 4 times greater than standard Diffie-Hellman key exchange that provides no authentication (i.e., no security against man-in-the middle attackers).

Very recently, Gennaro [34] provided ways of improving the efficiency of the KOY protocol and the GL protocol. They pointed out that both of the KOY and the GL protocols use one-time signatures to provide authentication (against man-in-the-middle attack) which increases the bandwidth requirement for the message transmission. They improve the efficiency of those protocols by replacing one-time signatures with faster and shorter message authentication codes. Consequently, assuming a security parameter of 128, such an improvement saves as much as 12 Kbytes of bandwidth; while one-time signature schemes require around 12Kbytes key and signature transmission, only 256 bits transmission is necessary for the MAC.

# B PAKE security implies forward security

In this section, we show that any password-based authenticated key exchange protocol secure in the standard model (or in the non-corruption model as opposed to the corruption model where password corruption is allowed) is also forward secure in weak-corruption model (where compromise of a password is allowed but complete control over users is not allowed). For details of those security definitions, we refer to [40, 10]. Recall that we only consider off-line dictionary attackers for forward security.

Before we give our proof, we briefly summarize the oracles used in the security definition of password-based key exchange [40, 10]:

- $\mathsf{Execute}(C, i, S, j)$ runs an execution of the protocol between new instances $\Pi_C^i$ (for user $C$) and $\Pi_S^j$ (for user $S$), and outputs the transcript of the protocol execution. This models passive eavesdropping.
- $\mathsf{Send}(U, i, M)$ sends message $M$ to instance $\Pi_U^i$ of user $U$ (if there is no instance $\Pi_U^i$, initiates a new one) and returns the response according to the protocol. This models active on-line attacks.
- $\mathsf{Corrupt}(U)$ outputs the password $pw$ of user $U$. This models password corruption.
- $\mathsf{Reveal}(\Pi_U^i)$ outputs the session key of instance $\Pi_U^i$ of user $U$ if $\Pi_U^i$ has a session key.
- $\mathsf{Test}(\Pi_U^i)$ is used to measure the adversarial power. A random bit $b$ is flipped; if $b = 0$, a random session key is returned. If $b = 1$, the session key of instance $\Pi_U^i$ of user $U$ is returned.

Theorem 1. *If a protocol $\pi$ is a secure password-based authenticated key exchange protocol in the non-corruption model, $\pi$ is forward secure against off-line attackers in the weak-corruption model.*

**Proof** Assume that there exists an adversary $\mathcal{A}$ breaking forward security of $\pi$ in the weak-corruption model. Because we consider only off-line dictionary attackers for forward security, $\mathcal{A}$ is not allowed to access Send oracle. Then, by the assumption, $\mathcal{A}$ attacks $\pi$ *in the weak corruption model* and succeeds in the experiment with probability $\frac{1}{2} + \delta(k)$, for a non-negligible function $\delta(k)$. Given $\mathcal{A}$, we can construct an adversary $\mathcal{A}'$ that attacks protocol $\pi$

*in the non-corruption model* by eavesdropping on the executions of $\pi$, and then outputs a pair of a password and a user, $(pw, U)$, for some user $U$ that $\mathcal{A}'$ has chosen, such that probability of $pw$ being $U$'s correct password is non-negligibly higher than $\frac{1}{|\mathcal{W}|}$. The existence of $\mathcal{A}'$ is sufficient to show that protocol $\pi$ is an insecure password-based authenticated key exchange protocol in the non-corruption model. The reason for this is as follows: informally, if there exists an adversary $\mathcal{M}$ who can correctly guess the password of any user of $\mathcal{M}$'s choice with probability non-negligibly higher than $\frac{1}{|\mathcal{W}|}$, then we can construct an on-line adversary $\mathcal{O}$ who uses $\mathcal{M}$ to break $\pi$ in the non-corruption model, with *one* on-line attack, and achieves an advantage non-negligibly higher than $\frac{1}{|\mathcal{W}|}$. Basically, $\mathcal{O}$ simulates the view of $\mathcal{M}$ and when $\mathcal{M}$ outputs $(pw, U)$ for some user $U$, $\mathcal{O}$ carries out an on-line attack against $U$ with password $pw$ and asks for a Test query for the instance. Given a challenge key as a response to a Test query, if the key is the same as the key that $\mathcal{O}$ computed in the on-line attack, $\mathcal{O}$ outputs 1. Otherwise, $\mathcal{O}$ outputs 0. Then, if $pw$ was a correct password for $U$, $\mathcal{O}$ always succeeds. Otherwise, $\mathcal{O}$ succeeds with probability exactly $\frac{1}{2}$. Therefore, the advantage of $\mathcal{O}$'s in breaking protocol $\pi$ in the non-corruption model with *one* on-line attack is non-negligibly higher than $\frac{1}{|\mathcal{W}|}$ (the advantage of $\mathcal{O}$ obtained by using $\mathcal{M}$ is the difference between the probability that $\mathcal{M}$ guesses a password correctly and the probability $\frac{1}{|\mathcal{W}|}$). Then, it leads a contradiction and the proof is complete.

Now, let's see how $\mathcal{A}'$ can guess a password of a user with probability non-negligibly higher than $\frac{1}{|\mathcal{W}|}$, by using $\mathcal{A}$. Adversary $\mathcal{A}'$, playing in the non-corruption model, has access to the Execute and Reveal oracles, and $\mathcal{A}$, playing in the weak-corruption model, has access to the Execute, Reveal, Corrupt and Test oracles. Adversary $\mathcal{A}'$ proceeds as follows:

1. Let $L$ be a maximum number of users that $\mathcal{A}$ will ask for Execute query ($L$ is polynomial in $k$ since $\mathcal{A}$ is a PPT adversary). Choose an integer $\ell$ from $\{1, ..., L\}$ at random.

2. Whenever $\mathcal{A}$ asks a query in a form of $\mathsf{Execute}(C, i, S, j)$,[13] if $C$ is the $\ell$-th new user that has been queried in such a form, keep $C$ as a target user $T$ and answer to $\mathcal{A}$ by forwarding the same query to its own oracle and returning the response from the oracle. Otherwise, choose a random password (or find a stored password for $C$, if one exists), and simulate an execution according to $\pi$, and return the resulting transcript to $\mathcal{A}$.

3. Whenever $\mathcal{A}$ asks a query in a form of $\mathsf{Corrupt}(U)$, if $U$ is not $T$, find a password chosen for $U$ and answer with it (if there is no record, answer with a random password and record it). Otherwise (if $U$ is $T$), choose a random password $pw_1$ and answer with $pw_1$.

4. Upon receiving a $\mathsf{Test}(\Pi^i_{U'})$ query from $\mathcal{A}$, adversary $\mathcal{A}'$ proceeds as follows:

   - If $U'$ is not $T$, $\mathcal{A}'$ selects a random password $pw$ and outputs $(pw, T)$. Let $\mathsf{NoUse}_{\mathcal{A}}$ denote this event.
   - Otherwise, if $U'$ is $T$, $\mathcal{A}'$ proceeds as follows:
     (a) Flips a random coin $b$. If $b = 0$, $\mathcal{A}'$ chooses a random session key $r$ and provides it to $\mathcal{A}$.
     (b) If $b = 1$, $\mathcal{A}'$ sends a $\mathsf{Reveal}(\Pi^i_{U'})$ query to its own oracle, obtains the real session key $\mathsf{sk}^i_{U'}$, and provides $\mathcal{A}$ with it.
     (c) If $\mathcal{A}$ aborts, $\mathcal{A}'$ chooses a random password $pw_2$, different from $pw_1$, and outputs $(pw_2, T)$. Let $\mathsf{Abort}_{\mathcal{A}}$ denote this event.
     (d) Finally, when $\mathcal{A}$ outputs $b'$, if $b' = b$, adversary $\mathcal{A}'$ outputs $(pw_1, T)$. Otherwise, $\mathcal{A}'$ chooses a random password $pw_3$, different from $pw_1$, and outputs $(pw_3, T)$.

Next, we analyze the probability that $\mathcal{A}'$ correctly guesses the password of a user. For a better understanding, we introduce some additional notation. We let $\mathsf{Succ}_{\mathcal{A}}$ (resp., $\mathsf{Succ}_{\mathcal{A}'}$) denote the event that $\mathcal{A}$ (resp., $\mathcal{A}'$) succeeds in the experiment of breaking forward security of $\pi$ (resp., guesses a correct password of a user). Also, we let $\mathsf{pw}_1$ (resp., $\mathsf{pw}_2$, or $\mathsf{pw}_3$) denote the event that $T$'s password is equal to $pw_1$ (resp., $pw_2$, or $pw_3$).

Then, the success probability of $\mathcal{A}'$ is the following:

$$\begin{aligned}\Pr[\mathsf{Succ}_{\mathcal{A}'}] &\geq \Pr[\mathsf{Succ}_{\mathcal{A}'}|\mathsf{NoUse}_{\mathcal{A}}] \times \Pr[\mathsf{NoUse}_{\mathcal{A}}] + \\ &\quad \Pr[\mathsf{Succ}_{\mathcal{A}'}|\overline{\mathsf{NoUse}_{\mathcal{A}}}] \times \Pr[\overline{\mathsf{NoUse}_{\mathcal{A}}}] \\ &\geq \frac{1}{|\mathcal{W}|} \times \left(1 - \frac{1}{L}\right) + \Pr[\mathsf{Succ}_{\mathcal{A}'}|\overline{\mathsf{NoUse}_{\mathcal{A}}}] \times \frac{1}{L} \\ &= \frac{1}{|\mathcal{W}|} + \frac{1}{L} \cdot \left(\Pr[\mathsf{Succ}_{\mathcal{A}'}|\overline{\mathsf{NoUse}_{\mathcal{A}}}] - \frac{1}{|\mathcal{W}|}\right) \quad (3)\end{aligned}$$

Then, we can bound the probability $\Pr[\mathsf{Succ}_{\mathcal{A}'}|\overline{\mathsf{NoUse}_{\mathcal{A}}}]$ as follows (*nb.*, for simplicity, we omit the conditional event $\overline{\mathsf{NoUse}_{\mathcal{A}}}$ for the right hand side) :

$$\begin{aligned}\Pr[\mathsf{Succ}_{\mathcal{A}'}|\overline{\mathsf{NoUse}_{\mathcal{A}}}] &= \Pr[\mathsf{pw}_2 \wedge \mathsf{Abort}_{\mathcal{A}}] + \\ &\quad \Pr[\mathsf{Succ}_{\mathcal{A}} \wedge \mathsf{pw}_1 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}] + \\ &\quad \Pr[\overline{\mathsf{Succ}_{\mathcal{A}}} \wedge \mathsf{pw}_3 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}]\end{aligned}$$

Next, we bound each term of the right hand side in the above equation. First, we bound the probability $\Pr[\mathsf{pw}_2 \wedge \mathsf{Abort}_{\mathcal{A}}]$

[13] $S$ is a server who keeps all the passwords of clients $C$.

as follows:

$$\begin{aligned}
\Pr[\mathsf{pw}_2 \wedge \mathsf{Abort}_{\mathcal{A}}] &= \Pr[\mathsf{pw}_2 \wedge \overline{\mathsf{pw}_1} \wedge \mathsf{Abort}_{\mathcal{A}}] \\
&= \Pr[\mathsf{pw}_2 \wedge \overline{\mathsf{pw}_1}] \times \Pr[\mathsf{Abort}_{\mathcal{A}} | \overline{\mathsf{pw}_1} \wedge \mathsf{pw}_2] \\
&= \Pr[\mathsf{pw}_2] \times \Pr[\mathsf{Abort}_{\mathcal{A}} | \overline{\mathsf{pw}_1}] \\
&= \frac{1}{|\mathcal{W}|} \times \Pr[\mathsf{Abort}_{\mathcal{A}} | \overline{\mathsf{pw}_1}] \\
&= \frac{1}{|\mathcal{W}|} \times \left(1 - \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \overline{\mathsf{pw}_1}]\right)
\end{aligned}$$

Second, we bound the probability $\Pr[\mathsf{Succ}_{\mathcal{A}} \wedge \mathsf{pw}_1 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}]$ as follows:

$$\begin{aligned}
\Pr[\mathsf{Succ}_{\mathcal{A}} \wedge \mathsf{pw}_1 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}] &= \Pr[\mathsf{Succ}_{\mathcal{A}} | \mathsf{pw}_1 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}] \times \\
&\quad \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \mathsf{pw}_1] \times \Pr[\mathsf{pw}_1] \\
&= \left(\frac{1}{2} + \delta(k)\right) \times \\
&\quad \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \mathsf{pw}_1] \times \frac{1}{|\mathcal{W}|}
\end{aligned}$$

Finally, the last term of probability $\Pr[\overline{\mathsf{Succ}_{\mathcal{A}}} \wedge \mathsf{pw}_3 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}]$ is bounded as follows:

$$\begin{aligned}
\Pr[\overline{\mathsf{Succ}_{\mathcal{A}}} \wedge \mathsf{pw}_3 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}] &= \Pr[\overline{\mathsf{Succ}_{\mathcal{A}}} | \mathsf{pw}_3 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}] \times \\
&\quad \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \mathsf{pw}_3] \times \Pr[\mathsf{pw}_3] \\
&= \left(1 - \Pr[\mathsf{Succ}_{\mathcal{A}} | \mathsf{pw}_3 \wedge \overline{\mathsf{Abort}_{\mathcal{A}}}]\right) \times \\
&\quad \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \overline{\mathsf{pw}_1}] \times \frac{1}{|\mathcal{W}|} \\
&\geq \frac{1}{2|\mathcal{W}|} \times \Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \overline{\mathsf{pw}_1}]
\end{aligned}$$

Let $p$ denote the probability $\Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \overline{\mathsf{pw}_1}]$ and $q$ denote the probability $\Pr[\overline{\mathsf{Abort}_{\mathcal{A}}} | \mathsf{pw}_1]$. Then, by combining three probabilities that we computed so far, we have:

$$\begin{aligned}
\Pr[\mathsf{Succ}_{\mathcal{A}'} | \overline{\mathsf{NoUse}_{\mathcal{A}}}] &\geq \frac{1}{|\mathcal{W}|} \times (1 - p) + \\
&\quad \left(\frac{1}{2} + \delta(k)\right) \times q \times \frac{1}{|\mathcal{W}|} + \frac{1}{2|\mathcal{W}|} \times p \\
&= \frac{1}{|\mathcal{W}|} + \frac{q}{|\mathcal{W}|} \times \delta(k) + \frac{(q - p)}{2|\mathcal{W}|} \qquad (4)
\end{aligned}$$

In the event of $\mathsf{pw}_1$, the simulated view for $\mathcal{A}$ is perfect. Therefore, the probability that $\mathcal{A}$ aborts in the conditional event of $\mathsf{pw}_1$ (i.e., $q$) is negligibly close to 1 (i.e., $q \approx 1$). Also, no matter how close the probability $p$ is to $q$, $q$ is greater than or equal to $p$. Therefore, by applying Equation (4) into Equation (3), we obtain:

$$\begin{aligned}
\Pr[\mathsf{Succ}_{\mathcal{A}'}] &\geq \frac{1}{|\mathcal{W}|} + \frac{1}{L} \times \frac{q}{|\mathcal{W}|} \times \delta(k) \\
&\geq \frac{1}{|\mathcal{W}|} + \frac{1}{2|\mathcal{W}|L} \times \delta(k)
\end{aligned}$$

which is non-negligibly higher than $\frac{1}{|\mathcal{W}|}$ since $L$ is polynomial in $k$. This completes the proof. ■