Formal Analysis of Distance Bounding with Secure Hardware

Handan Kılınç and Serge Vaudenay

EPFL, Lausanne, Switzerland

Abstract. A distance bounding (DB) protocol is a two-party authentication protocol between a prover and a verifier which is based on the distance between the prover and the verifier. It aims to defeat threats by malicious provers who try to convince that they are closer to the verifier or adversaries which seek to impersonate a far-away prover. All these threats are covered in several security definitions and it is not possible to have a single definition covering all. In this paper, we describe a new DB model with three parties where the new party is named hardware. In this model, called secure hardware model (SHM), the hardware is held by the prover without being able to tamper with. We define an all-in-one security model which covers all the threats of DB and an appropriate privacy notion for SHM. In the end, we construct the most efficient (in terms of computation by the prover-hardware and number of rounds) and secure DB protocols achieving the optimal security bounds as well as privacy.

Keywords: distance bounding, RFID, NFC, relay attack, tamper resistance, terrorist fraud

1 Introduction

Distance bounding (DB) protocols are two-party (a prover and a verifier) authentication protocols. A prover authenticates itself and proves that the distance between its location and the verifier's location is less than a predetermined bound. The verifier estimates the distance of the prover by computing the round trip time of sending a challenge to the prover and receiving a response from the prover. Brands and Chaum [6] first defined this notion as a solution to relay attacks. However, it may also provide a solution for the following threats:

Mafia Fraud (MiM) [12] : A man-in-the-middle (MiM) adversary between a verifier and a far-away honest prover makes the verifier accept the access of the prover. Malicious and far-away provers who want to convince that they are close to a verifier appear in the followings:

Distance Fraud (DF) : A malicious far-away prover tries to prove that he is close enough to the verifier to make the verifier accept.

Distance Hijacking (DH) [11] : A far-away malicious prover takes advantage of some honest and active provers who are close to the verifier to make the verifier grant privileges to the far-away prover.

Terrorist fraud (TF) [12] : A far-away malicious prover, with the help of an adversary, tries to make the verifier accept the access of the prover.

Clearly, the strongest security notion is the resistance to TF. So, if we can construct a DB protocol that is secure against TF, then the DB protocol will be secure against MiM, DF and DH. However, it is not possible to achieve the TF-security because of a trivial attack: the malicious prover gives his secret (key) to a close adversary, and the adversary authenticates on behalf of the malicious prover by running the protocol. To achieve the TF-security, the trivial attack is artificially excluded from the TF model in the literature by assuming that malicious provers would never share their keys (in this paper, we call this weaker version "TF'-security"). However, we cannot adapt TF'-security as an all-in-one security notion because no connection between TF'-security and MiM, DF or DH security can be established. Because of this disconnection, all DB protocols require separate security analysis for each of them. The only public-key DB protocols that are secure against all of them (MiM, DF, DH, TF') are ProProx [27], its variant eProProx [25] and TREAD [2]. Some important distance bounding protocols [6, 8, 10, 15, 21, 23, 19] are all vulnerable to TF'. The protocol by Bultel et al. [7] is TF'-secure thanks to a 'cheat option' (as explained below) but it is not DH-secure since it aims for anonymity against verifier.

Moreover, the formal definition of TF'-security is controversial. The TF'-security definition of Dürholz et al. [13] allows treatment of the partial disclosure of the secret key. Essentially, the TF' security in this definition implies that any information forwarded to a close-by adversary would allow another adversary to later pass, without a help of the prover, with the same probability, but through a "cheat option" in the protocol. Fischlin and Onete [14] adapted the Swiss-Knife protocol [20] to have this definition. However, it was proven that this technique weakens Swiss-Knife for MiM-security [24]. Clearly, it is not reasonable to weaken the most relevant security to protect it against the least relevant one. There are also extractor based TF'-security definitions [5, 27, 24] stronger than the definition of Dürholz et al. model [13]. However, all TF'-security definitions are constructed with the assumption that the malicious prover do not reveal any secret key related information. This assumption is considered **weak and not realistic** [1]. In short, none of the models in the literature fully covers TF.

Apparently, there is no way of achieving TF-security without hiding the secret key from the prover. This intuitive idea has been noticed [22, 9], but never formally defined. A natural question to ask here is whether this idea really prevents TF. The answer is "yes and no" because hiding the key is necessary but not sufficient.

In a nutshell, state of the art DB results says that TF-security is not possible in the existing models of DB and it could be possible by hiding the key but this is not enough. However, it is still not formally noted how it can be achievable.

Therefore, in this paper, we define a new formal model where constructing TF-secure protocols are possible.

Our formal model for DB, which we call secure hardware model (SHM), provides a solution to all DB related problems that we mention. We denote the two-algorithm (Prover and Verifier) DB corresponding to the classical DB in the literature as "plain model" (PM) [6, 13, 4, 5, 26]. In the SHM, we have another entity called "Hardware" that is always honest and only communicate with their holder (the prover). Mainly, this hardware runs some part of the prover algorithm honestly and neither a malicious prover nor an adversary can corrupt it. In the real world, we can realize our new entity as e.g. tamper-resistant modules in smart-cards. In more detail, *our contribution* in this paper is the following:

- We define a new type of DB with three algorithms (V, P, H): verifier, prover, hardware. Then, we design a communication and adversarial model for three-algorithm DB which we call secure hardware model (SHM). In SHM, it is possible to have TF-secure DB protocols without excluding trivial attacks. We give a new security definition in SHM for a three-algorithm DB. In this security definition, achieving TF-security means achieving MiM, DF and DH-security. So, we obtain an all-in-one definition.
- We obtain a convincing model for TF based on SHM. We show that the TF-security of (V, P, H) in SHM is equivalent to the MiM-security of (V, H) in PM where H in PM corresponds to the prover algorithm. This result implies that ***P* plays no role in security but only in the correctness of the protocol to have TF-security**.
- We establish security relations between PM and SHM. We show that the MiM-security in SHM and the MiM-security in PM are equivalent where the prover algorithm in PM is the union P^H of the prover P and the hardware H in SHM. Additionally, we show that a MiM-secure DB protocol in PM can be converted into a fully-secure DB protocol in SHM. This result shows that if we have only a MiM-secure DB protocol in PM, **we can easily construct an efficient DB protocol secure against all threats in SHM**.
- We define a strong privacy notion of DB in SHM. Strong privacy in DB requires that the adversary cannot identify a prover even after getting his secret (e.g. by a corruption).
- We construct a symmetric DB protocol **MiM-symDB which is the most efficient optimally secure MiM-secure protocol in PM (in terms of computation and number of rounds) among the protocols with binary challenges and responses.** Then, we convert it into a DB protocol in SHM (Full-symDB^H) and obtain the most efficient symmetric DB protocol secure against all threats and achieving optimal security bounds.
- We also consider a secure and private public-key DB protocol in SHM. Instead of designing a new one we take advantage from existing public-key DB's Eff-pkDB and Simp-pkDB [19] to convert them into SHM. We slightly modify Eff-pkDB to increase its efficiency in SHM, and we modify Simp-pkDB such that its new version in SHM is secure and private.

We underline that the **only assumption on the secure hardware is that it is honest** which means that it runs the specified algorithm only. By doing so, we give a model here where the TF-security is achievable.

One may argue that our assumption on secure hardware is too strong for the real world applications. For example, in the real world, if the secure hardware is implemented using a tamper-resistant hardware, it is always possible that a side-channel attack will break our assumption. However, we believe that relying on our assumption is more reasonable than relying on some adversarial intention (e.g., that the adversary never shares his secret). We can never prevent a TF-adversary to share his secret-key, but we can construct a strong tamper-resistant hardware which requires very expensive devices to be tampered. Besides, MiM-security would be preserved even if the tamper resistance assumption is broken.

2 Definitions and Security in SHM

We first give the formal definitions of SHM and security in this model. Then, we provide some security relations related to PM and SHM.

2.1 Definitions

Parties of a DB protocol are a prover and a verifier [6]. However, we define a new version of it called three-algorithm (symmetric or public-key) DB where the algorithms are prover, verifier, and hardware.

Definition 1 (Three-Algorithm Symmetric DB). *Three-algorithm symmetric DB is a probabilistic polynomial-time (PPT) protocol. It consists of a tuple $(\mathcal{K}, V, P, B, H)$ where $\mathcal{K}$ is the key generation algorithm, P is the proving algorithm, H is the hardware algorithm, V is the verifying algorithm and B is the distance bound. The input of V and H is K generated by $\mathcal{K}$. P interacts with $H(K)$ and $V(K)$. At the end of the protocol, $V(K)$ outputs a final message $\mathsf{Out}_V \in \{0,1\}$. If $\mathsf{Out}_V = 1$, then V accepts. If $\mathsf{Out}_V = 0$, then V rejects.*

In symmetric DB, V knows that it needs to use K (possibly resulting from a prior identification protocol).

Definition 2 (Three-Algorithm Public key DB). *Three-algorithm public key distance bounding is a PPT protocol. It consists of a tuple $(\mathcal{K}_P, \mathcal{K}_V, V, P, B, H)$ where $(\mathcal{K}_P, \mathcal{K}_V)$ are the key generation algorithms of P and V, respectively. The output of $\mathcal{K}_P$ is a secret/public key pair $(\mathsf{sk}_P, \mathsf{pk}_P)$ and the output of $\mathcal{K}_V$ is a secret/public key pair $(\mathsf{sk}_V, \mathsf{pk}_V)$. V is the verifying algorithm with the input $(\mathsf{sk}_V, \mathsf{pk}_V)$, P is the proving algorithm with the input $(\mathsf{pk}_P, \mathsf{pk}_V)$ and H is the hardware algorithm with the input $(\mathsf{sk}_P, \mathsf{pk}_P)$. B is the distance bound. P interacts with $H(\mathsf{sk}_P, \mathsf{pk}_P)$ and $V(\mathsf{sk}_V, \mathsf{pk}_V)$. At the end of the protocol, $V(\mathsf{sk}_V, \mathsf{pk}_V)$ outputs a final message $\mathsf{Out}_V \in \{0,1\}$ and has pk_P as a private output. If $\mathsf{Out}_V = 1$, then V accepts. If $\mathsf{Out}_V = 0$, then V rejects.*

This definition assumes a priori identification of pk_V for P.

Definition 3 (Correctness of DB). *A public-key (resp. symmetric) DB protocol is correct if and only if under an honest execution, whenever the distance between P and V is at most B, V always outputs $\mathsf{Out}_V = 1$ and pk_P (resp. $\emptyset$).*

In all definitions below, verifiers, provers, and hardware are parties running V, P and H, respectively. The parties can move and run their algorithms multiple times. Each new execution of a party's algorithm is an **instance** of this party.

Classical DB in the literature is very similar to three-algorithm DB with the following differences: no H algorithm exists and the input of P in public-key and symmetric DB is $(\mathsf{sk}_P, \mathsf{pk}_P, \mathsf{pk}_V)$ and K, respectively. The plain model is the model corresponding to the classical DB.

Plain Model (PM): Parties of PM are provers, verifiers and other actors.
- Honest parties run their assigned algorithms only.
- Verifiers are always honest. Provers are either malicious or honest.
- Each instance of a party has a location.
- A malicious party may run its instances concurrently, while an honest party runs its instances only sequentially.
- Communication between instances has a latency proportional to the distance (e.g., it travels at the speed of light).

The secure hardware model is the model corresponding to three-algorithm DB: P, V and H.

Secure Hardware Model (SHM): Parties of SHM are provers, hardware, verifiers and other actors. SHM includes all the characteristics of PM and the additional ones:
- Secure hardware are honest parties.
- Each prover possesses its own secure hardware.
- The secure hardware of an honest prover can only communicate with its prover and they are both at the same location.

In the rest of the paper, whenever we say "a distance bounding protocol in SHM", it refers to the three-algorithm DB.

Remark that since secure hardware are honest parties, they always run their assigned algorithms even if malicious provers hold them. They should be taken as a subroutine of a prover algorithm running on a secure enclave where the prover can never change or interfere it.

SHM and PM follow the communication model from [5, 18]: instances can only communicate by sending messages which are delivered with a delay proportional to the distance, and malicious instances can prevent delivery or change the destination of messages [27].

Now, we give our security definition for a DB protocol in SHM. The definition covers distance fraud, mafia fraud (MiM), distance hijacking and terrorist fraud which are the threat models in PM.

Definition 4 (Security in SHM). *Consider a public-key DB. The game consists of a verifier and provers $P_1, P_2, ..., P_t$ with their corresponding hardware $H_1, H_2, ..., H_t$. It begins by running the key setup algorithm $\mathcal{K}_V$ outputting*

$(\mathsf{sk}_V, \mathsf{pk}_V)$ *for* V *and* $\mathcal{K}_P$ *outputting* $(\mathsf{sk}_{P_i}, \mathsf{pk}_{P_i})$ *for* H_i*. The game consists of instances of the verifier, provers, hardware and actors.* $\mathcal{V}$ *is a distinguished instance of the verifier. One prover (let's denote* P*) is the target prover. The winning condition of the game is* $\mathcal{V}$ *outputs* $\mathsf{Out}_V = 1$ *and privately* pk_P *(public key of* P*) if no close instance of* P*'s hardware exists during the execution of* $\mathcal{V}$*.*

- *The DB protocol is MiM-secure if the winning probability is always negligible whenever* P *is honest*[1]*.*
- *The DB protocol is DF-secure if the winning probability is always negligible whenever there is no instance of any party close to* $\mathcal{V}$*.*
- *The DB protocol is DH-secure if the winning probability is always negligible whenever all close instances are honest provers other than* P *and their hardware.*
- *The DB protocol is TF-secure if the winning probability is always negligible.*

The same security definition holds for a symmetric DB where we replace $\mathcal{K}_V$ *and* $\mathcal{K}_P$ *with* $\mathcal{K}$ *and* $\mathsf{sk}_{P_i}/\mathsf{pk}_{P_i}$ *with* K_i*.*

Without loss of generality, we can consider all other actors as adversaries.

It is clear that TF-security implies DF-security, MiM-security, and DH-security. So, we have an all-in-one security notion in SHM. Hence, we say **"secure" instead of "TF-secure"** in SHM.

Security in PM: The security in PM is almost the same as Definition 4 except that in PM, we do not have hardware. In PM, there is always a trivial TF-attack in which a malicious prover can give his secret key to another malicious party so that the party authenticates the prover while it is far-away. So, TF-security is not possible in PM. Clearly, this trivial attack is preventable in SHM if we can assure that H never leaks K.

Note that we do not consider the weaker version of TF-security [13, 20, 24] (TF'-security) which artificially excludes trivial attack. So, when we refer to TF-security in PM, we indeed refer to an impossible-to-achieve notion.

Notations:

$\boldsymbol{P_{dum}}$ is a dummy prover algorithm in SHM which only relays the messages between the outside world and H without even using any of its input. Remark that if the prover who should run P_{dum} is malicious, then it can still play with its hardware or other parties maliciously.

$\boldsymbol{P^H}$ is the algorithm which is constructed from joining P and H in SHM. More precisely, P^H runs P and instead of interacting with H, it executes the same computation that H would do if P had interacted. Therefore, $\boldsymbol{P^H_{dum}}$ is the hardware algorithm H.

"Challenge phase" is informally defined as the phase where V comprehends the proximity of P. In the challenge phase, V sends challenges and receives responses from P. If all the responses are correct and arrive on time, then V decides that the distance between V and P is less than the bound B.

[1] Recall that it implies that H communicates with only P and that they are at the same location.

2.2 Security Results

We give some security relations between a DB protocol in PM and SHM.

Theorem 1 (MiM in SHM $\Rightarrow$ MiM in PM). *Let $DB = (\mathcal{K}, V, P, B, H)$ be a symmetric-key DB protocol in SHM. We define a DB protocol $DB' = (\mathcal{K}, V, P^H, B)$ in PM. If DB is MiM-secure then DB' is MiM-secure.*
The same holds with public-key DB.

The proof is trivial by adding a hardware to every honest prover at the same location: A MiM-game against DB' becomes a MiM-game against DB.

Theorem 2 (MiM-security in PM with P^H_{dum} $\Leftrightarrow$ Security in SHM). *Let $DB = (\mathcal{K}, V, P, B, H)$ be a symmetric DB in SHM and and $DB' = (\mathcal{K}, V, P^H_{dum}, B)$ be a symmetric-key DB in PM where H in DB' corresponds H of DB. DB' is MiM secure in PM if and only if DB is TF-secure in SHM.*

Here, the prover algorithm of DB' is just H because $P^H_{dum} \equiv H$.

Note that DB' in Theorem 2 is not a correct DB protocol in general if $P \neq P_{dum}$ as the algorithm P disappeared. However, we can still consider MiM-security for DB' without correctness.

Proof. ($\Rightarrow$) Consider a TF-game in SHM. We run this game in PM by simulating the secure hardware H of DB with the prover P^H_{dum} of DB' and simulating the prover P in SHM with an actor in PM (it is possible because P in SHM does not have any secret key as an input). Then, we obtain MiM-game of DB'.

($\Leftarrow$) If $\mathcal{A}$ wins the MiM-game of DB', then a TF adversary runs $\mathcal{A}$ and wins the TF-game for DB. $\square$

Remark that it is not possible to prove "MiM-security of $DB' = (\mathcal{K}, V, P^H, B)$ $\Leftrightarrow$ security of $DB = (\mathcal{K}, V, P, B, H)$" where P in DB' is not necessarily P_{dum} because we could not simulate H and P in "$\Rightarrow$" case of the proof in Theorem 2.

Theorem 2 clearly shows that hiding the key is necessary to have security in DB. Because if H in DB does not hide the key, then the prover algorithm of DB' which is H does not hide as well. So, DB' would not be MiM-secure. However, hiding may not be enough as explained below:

We consider the algorithm P of DB which does the computations $\mathcal{C} = \{C_1, C_2, ..., C_k\}$ and where P does learn any key related information. So, all computations in $\mathcal{C}$ are executed independently from the key. Assume that the success probability of an adversary to break the TF-security of DB is at most p. Then, thanks to Theorem 2, the success probability of a MiM-adversary $\mathcal{A}$ in DB' is at most p. In addition, assume that there exists $C_i \in \mathcal{C}$ and the success probability of a TF-adversary in DB is $p' > p$ without C_i. If such C_i exists, we can have another MiM-adversary $\mathcal{A}'$ which runs P without C_i [2] and wins

[2] Remark that any adversary can compute the computations in $\mathcal{C}$ because they do not require any secret.

MiM-game with $p' > p$. So, this contradicts with our assumption which says that the success probability of a MiM-adversary can be at most p. As a result, Theorem 2 actually shows that the TF-security is not possible in SHM if the computations of P has an effect on p.

We agree that having a secure hardware running whole algorithm without its prover's effect on the security is a trivial solution to have TF-security. However, here, we show that **the other way around is not possible.** We underline that it does not mean that prover cannot do any computation to have TF-security. For example, in our TF-secure protocols in Section 4, the prover algorithm in SHM still executes some part of the algorithm P^H in PM but it does not have any effect on the security of the protocol (as it can be seen in their security proofs Theorem 8 and Theorem 10.)

Some more results of Theorem 2:

- We can conclude if $DB' = (\mathcal{K}, V, P^H_{dum}, B)$ is **MiM-secure** and correct DB protocol, then we can construct a **secure** DB protocol $DB = (\mathcal{K}, V, P, B, H)$ in SHM for any algorithm P. DB is further correct when $P = P_{dum}$.
- In order to prove security of $DB = (\mathcal{K}, V, P, B, H)$ in SHM, it is enough to prove MiM-security of $DB' = (\mathcal{K}, V, P^H_{dum}, B)$ in PM.
- **MiM security and security of a DB protocol $\boldsymbol{DB = (\mathcal{K}, V, P, B, H)}$ in SHM are equivalent if $\boldsymbol{P = P_{dum}}$** due to Theorem 1 and Theorem 2. Note that this result may not hold without P_{dum} .

In Figure 1, we give the security (non)-implications in SHM and PM. The proof of these (non)-implications are in the full version of the paper. In Figure 2, we give the same for SHM when the prover is P_{dum}. In this case, **the full security is equivalent to MiM-security**. The rest of the (non)-implications in Figure 2 can be proven the same as in the non-implications in Figure 1.

$$\begin{tikzcd}
\mathsf{TF} \arrow[rr, shift left] \arrow[dr, shift left] & & \mathsf{DH} \arrow[ll, shift left, "\times" marking] \arrow[rr, shift left] \arrow[d, shift left, "\times" marking] & & \mathsf{DF} \arrow[ll, shift left, "\times" marking] \arrow[dl, "\times" marking] \\
& \mathsf{MiM} \arrow[ul, shift left, "\times" marking] \arrow[ur, "\times" marking] \arrow[urrr, "\times" marking] & & &
\end{tikzcd}$$

Fig. 1. Security implications of DB protocols in PM and SHM. TF-security implies all of them, DH-security implies DF security and no relation exists between MiM and DH (also DF).

$$\begin{tikzcd}
\mathsf{TF} \equiv \mathsf{MiM} \arrow[r, shift right] & \mathsf{DH} \arrow[l, shift right, "\times" marking] \arrow[r, shift right] & \mathsf{DF} \arrow[l, shift right, "\times" marking]
\end{tikzcd}$$

Fig. 2. Security implications in SHM with the prover P_{dum}. TF-security and MiM security are equivalent in SHM with P_{dum}. The relations between DF, DH and MiM are the same as in Figure 1.

2.3 Privacy

In strong-privacy definition of PM, the adversary can corrupt the provers and learn the secrets. However, the hardware in SHM is honest by nature. So, it cannot be corrupted. Hence, we define semi-strong privacy with no such corruption. Achieving semi-strong privacy in a DB protocol is good enough assuming that the hardware is tamper-resistant. Nevertheless, we also allow corruption of hardware in order to define the strong privacy notion.

Definition 5 (Privacy in SHM). *The privacy game consists of a verifier, provers $P_1, P_2, ..., P_t$ and their corresponding hardware $\{H_1, H_2, ..., H_t\}$. We generate the secret/public key pairs of them with $\mathcal{K}_V$ and $\mathcal{K}_P$ for the verifier and the hardware of provers. We pick $b \in \{0, 1\}$ and start the game:*
The adversary can create instances of the verifier and any prover. It can send/receive messages to/from instances of the verifier. It can corrupt any prover and hardware which let it learn the current state of their memory. At some moment, it picks two provers P_i, P_j as a challenge of the game. If $b = 0$, we create a virtual prover of P_i with its hardware and if $b = 1$, we create a virtual prover of P_j with its hardware. The adversary can communicate with the virtual prover and its virtual hardware. It can also release a virtual prover, if it exists. In this case, we remove the virtual prover from the game, anonymously. The game has to have at most one virtual prover. In the end, the adversary outputs b'. If $b' = b$, the adversary wins. Otherwise, it loses.

We say a DB protocol in SHM is ***strong private*** *if the advantage of the adversary in this game is bounded by a negligible probability. We say a DB protocol in SHM is* ***semi-strong private*** *if the advantage of the adversary in a version of this game, where the corruption only lets the adversary communicate with the hardware non-anonymously, is bounded by a negligible probability.*

In semi-strong privacy, even though we do not allow corruption of hardware, we let semi-strong corruption occur by allowing interaction with the secure hardware. In SHM, we stress that when P interacts with its secure hardware, this interaction remains private.

Hermans et al. [16] defined a similar game for the **strong privacy** of DB in PM. In that game, no hardware exists, so the definition of semi-strong privacy is not considered. Instead, the **weak privacy** notion exists where no corruption on provers are allowed.

Note that we obtain a notion of strong privacy of $DB = (K, V, P, B, H)$ in SHM which is equivalent to the strong privacy of $DB' = (K, V, P^H, B)$ in PM.

3 Optimal symmetric DB protocol in SHM

In this section, we show our new protocol MiM-symDB in PM which is only MiM-secure (not DF, DH or TF-secure). We construct a DB at this level of security because having MiM-security in PM is enough to achieve (full) security in SHM as a result of Theorem 2. The security bounds of MiM-symDB is very close to optimal security bounds [5] [3]. Its conversion into SHM reaches the same

[3] A security bound of a DB means an upper bound of the success probability of an adversary.

bound as well. It is proved [5] that an optimal security bound in PM for a MiM-adversary is $(\frac{1}{2})^n$ given that challenges and responses are bits and the challenge phase consists of n rounds. The same bound applies in SHM as well.

We note that using other optimally MiM-secure DB protocols such as DB1, DB2, DB3 [5] is reasonable as well to have fully secure DB protocols in SHM. However, these protocols are also secure against DF or TF' in PM which is an overkill since we need only MiM-security. By constructing an optimal MiM-only secure DB in PM, we can save some computations and rounds.

Notation: When we use H as a superscript in the name of a protocol, it shows that it is in SHM.

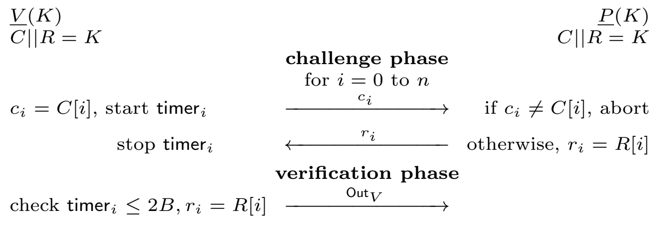

Fig. 3. MiM-OTDB

MiM-OTDB: First, we describe our MiM-OTDB protocol which is MiM-secure when it is executed *only once*. The prover P and the verifier V share a secret key $K = C||R$. Here, the bits of C correspond to the challenges and the bits of R correspond to the responses. In the challenge phase, in each round i, V sends the challenge $c_i = C[i]$ to P and P sends the response $r_i = R[i]$ to V. If P receives a challenge which is different from $C[i]$, then P does not continue the protocol. In the verification phase, V checks if the responses are correct and on time. (See Figure 3.)
MiM-symDB: The prover P and the verifier V share a secret key K. They use a pseudo random function (PRF) f returning strings of $2n$ bits. P and V exchange the nonces $N_P, N_V \in \{0,1\}^s$, respectively, where s is a security parameter. Then, P and V compute $f(K, N_P, N_V)$ which outputs $C||R$. Finally, V and P run MiM-OTDB with using $C||R$ as a key. (See Figure 4.)

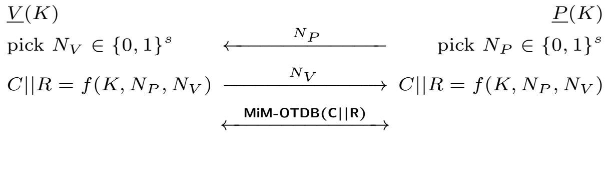

Fig. 4. MiM-symDB

We prove the following theorem by using a lemma in [18]. The lemma shows that any message m sent by a party is independent from the messages seen by another party at the time which is less than arrival time of m. This lemma is correct in SHM as well.

Theorem 3 (MiM-security of MiM-symDB). *If f is a secure PRF, then the winning probability of a probabilistic polynomial time (PPT) adversary in a MiM-game of MiM-symDB in PM is at most $\frac{3}{2^{n+1}} + \frac{q^2}{2^{s+1}} + \frac{q'^2}{2^{s+1}} + \mathsf{Adv}_{PRF}(q+q',t)$. For a PPT game, this is negligible.*

Here, q is the number of prover instances, q' is the number of verifier instances, t is the total complexity of the game and Adv_{PRF} is the advantage for distinguishing the output of f from the output of a random function with $q+q'$ queries and complexity t.

Proof. Γ_0 : It is a MiM-game where P's instances and V's instances with the distinguished instance $\mathcal{V}$ play in PM. The winning probability in Γ_0 is p.

Γ_1 : We reduce Γ_0 to Γ_1 where the nonces of the prover instances and the nonces of the verifier instances do not repeat. The probability that a prover (resp. verifier) instance selects the same nonce with the one of the other prover (resp. verifier) instances is bounded by $\frac{q^2}{2}\frac{1}{2^s}$ (resp. $\frac{q'^2}{2}\frac{1}{2^s}$). So, the winning probability of Γ_1 is at least $p - \frac{q^2}{2^{s+1}} - \frac{q'^2}{2^{s+1}}$.

Γ_2 : We reduce Γ_1 to Γ_2 where $\mathcal{V}$ and the prover's instances replace $f(K,.,.)$ by a random function. Clearly, the winning probability in Γ_2 is at least $p - \frac{q^2}{2^{s+1}} - \frac{q'^2}{2^{s+1}} - \mathsf{Adv}_{PRF}(q+q',t)$.

In Γ_2, we have a game where at most one prover instance P seeing (N_P, N_V) pair with $\mathcal{V}$ and $C||R$ is completely random meaning that it is independent from N_P and N_V. If P exists, it has to be far from $\mathcal{V}$ because of the winning condition of MiM-game. Assuming that $\mathcal{V}$ and P see the same (N_P, N_V), we look each round i for the case where r_i arrived on time. If r_i arrived on time, thanks to the lemma in [18], the response sent by P is independent from r_i or the challenge that P received is independent from c_i sent by $\mathcal{V}$. In any case, the adversary's probability to pass each round is $\frac{1}{2}$ because the response r_i has to be correct and on time: the adversary guesses either r_i or c_i (post-ask or pre-ask attack). There may also be one round where the pre-ask strategy is done for a constant number of rounds until it makes P abort. After abort, there is an additional opportunity (in the last of these rounds) for the adversary to pass the round by guessing the response. Therefore,

$$p = \frac{3}{2^{n+1}} + \frac{q^2}{2^{s+1}} + \frac{q'^2}{2^{s+1}} + \mathsf{Adv}_{PRF}(q+q',t).$$

□

Assuming that $\frac{q^2}{2^{s+1}} + \frac{q'^2}{2^{s+1}} + \mathsf{Adv}_{PRF}(q+q',t)$ is negligible, the success probability of a MiM-adversary is $\mathbf{\frac{3}{2^{n+1}}}$ very close to the optimal security $\frac{1}{2^n}$.

MiM-symDB is more efficient than the existing optimally MiM-secure protocols DB1, DB2, DB3 [5]. P in DB1, DB2, DB3 compute a

PRF function two times and some other mappings too. So, with parameter $n_c = n_r = 2$ in common structure, for a given target security, we construct a nearly optimal protocol, both in terms of number of round and computation complexity.

Theorem 4 (OT-MiM security of MiM-OTDB). *Any MiM-game against MiM-OTDB with **only one instance** of $V(K)$ and one instance of $P(K)$ has a winning probability bounded by $\frac{3}{2^{n+1}}$. In short, MiM-OTDB is OT-MiM-secure (one time MiM-secure) [26].*

Proof. Using the last game in the proof of Theorem 3, we can show that MiM-OTDB is OT-MiM-secure. □

MiM-OTDB is the most efficient one-time MiM-secure protocol [26] since it does not need any computation.
Adaptation of MiM-symDB to SHM (Full-symDB^H): We define Full-symDB^H with the tuple $(\mathcal{K}, V, P_{dum}, B, H)$ where B, V and $\mathcal{K}$ are as in MiM-symDB, H is the same with P in MiM-symDB.

Theorem 5 (Security of Full-symDB^H). *If f is a secure PRF, Full-symDB^H is secure in SHM.*

Proof. The conversion of Full-symDB^H in PM is $(\mathcal{K}, V, P^H_{dum}, B)$ which is equal to MiM-symDB. We know that MiM-symDB is MiM-secure since f is a secure PRF. Hence, Full-symDB^H with $(\mathcal{K}, V, P_{dum}, B.H)$ is secure thanks to Theorem 2. The security bound of Full-symDB^H is the same as MiM-symDB's. □

Full-symDB^H is the first protocol that reaches the optimal secure bounds for MiM, DH, DF and TF secure.

4 Optimal Public-key DB Protocols in SHM

In this section, we give two public key DB protocols in SHM: Full-pkDB1^H and Full-pkDB2^H[4] which is correct, private and secure. The first one is derived from Simp-pkDB [19] in PM. We modify Simp-pkDB to make it private in SHM because Simp-pkDB is not private in PM. The second one is derived from Eff-pkDB^p [19] in PM which is a variant of Eff-pkDB [19] with privacy protection. We slightly modify Eff-pkDB^p as well as Eff-pkDB to increase its efficiency. We use these protocols because of their efficiency in PM.
Full-pkDB1^H: This protocol is derived from Simp-pkDB [19]. However, Simp-pkDB is not private. Therefore, we add an extra encryption process on the prover side to achieve privacy. The details of Full-pkDB1^H is as follows:

The input of the verifier V is its secret/public key pair $(\mathsf{sk}_V, \mathsf{pk}_V)$ which is generated from the key generation algorithm of an encryption scheme $(\mathsf{Enc}', \mathsf{Dec}')$. The input of H is the prover's secret/public key pair $(\mathsf{sk}_P, \mathsf{pk}_P)$ which is generated by the key generation algorithm of the encryption scheme

[4] Full refers full security (MiM, DF, DH, TF) and privacy.

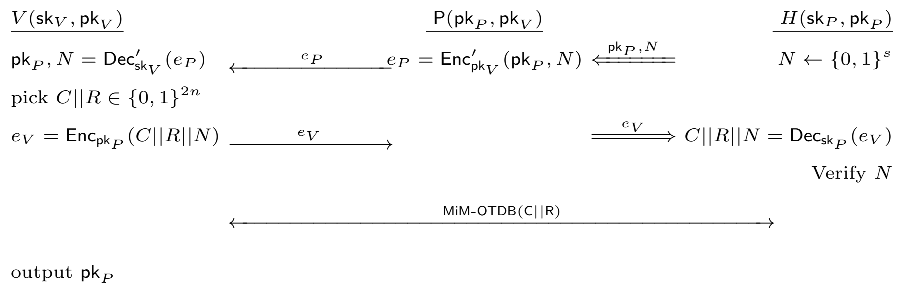
$V(\mathsf{sk}_V, \mathsf{pk}_V)$
$\mathsf{P}(\mathsf{pk}_P, \mathsf{pk}_V)$
$H(\mathsf{sk}_P, \mathsf{pk}_P)$
$\mathsf{pk}_P, N = \mathsf{Dec}'_{\mathsf{sk}_V}(e_P)$
e_P
$e_P = \mathsf{Enc}'_{\mathsf{pk}_V}(\mathsf{pk}_P, N)$
pk_P, N
$N \leftarrow \{0,1\}^s$
pick $C||R \in \{0,1\}^{2n}$
$e_V = \mathsf{Enc}_{\mathsf{pk}_P}(C||R||N)$
e_V
e_V
$C||R||N = \mathsf{Dec}_{\mathsf{sk}_P}(e_V)$
Verify N
MiM-OTDB(C||R)
output pk_P

Fig. 5. Full-pkDB1^H. The double arrow shows the communication between P and H

$(\mathsf{Enc}, \mathsf{Dec})$. The input of P is $(\mathsf{pk}_P, \mathsf{pk}_V)$. H picks a nonce N from $\{0,1\}^s$ and sends it to P along with pk_P. Then, P encrypts pk_P, N with pk_V and sends the encryption e_P to V. V learns pk_P and N by decrypting e_P with sk_V. Then, it picks $C||R$ from $\{0,1\}^{2n}$ and encrypts $C||R||N$ with pk_P. Next, it sends the encryption e_V to P and P relays it to H. H decrypts e_V and learns $C||R||N$. If N is the same nonce that it picked, it runs MiM-OTDB($C||R$) with V. The protocol is depicted in Figure 5.

The conversion of Full-pkDB1^H into PM is called as "Simp-pkDB^p". Its prover algorithm is $P^{H(\mathsf{sk}_P, \mathsf{pk}_P)}$ where P and H is from Full-pkDB1^H. Simp-pkDB^p is the same as Simp-pkDB except that the prover encrypts its public key and the nonce, and the verifier learns the public key and the nonce via decryption. Clearly, Simp-pkDB^p is MiM secure since Simp-pkDB is MiM-secure [19].

Theorem 6 (Security of Full-pkDB1^H). *If the encryption scheme (*$\mathsf{Enc}, \mathsf{Dec}$*) is IND-CCA secure and MiM-OTDB is OT-MiM-secure, Full-pkDB1^H is secure in SHM.*

Proof. Consider $DB = (\mathcal{K}_V, \mathcal{K}_P, V, P^H_{dum}, B)$ with V and H from Full-pkDB1^H. Actually, DB = Simp-pkDB. Using Theorem 2, Full-pkDB1^H is secure because DB = Simp-pkDB is MiM-secure [19] assuming that $(\mathsf{Enc}, \mathsf{Dec})$ is IND-CCA secure and MiM-OTDB is OT-MiM-secure.

□

Full-pkDB1^H achieves almost optimal security bounds because MiM-security of Simp-pkDB is reduced to MiM-security of MiM-OTDB [19].

We see that Full-pkDB1^H is secure without encryption. Actually, the encryption is only used for achieving privacy. So, if privacy is not a concern, we can use Full-pkDB1^H without the encryption and decryption. In this case, the verifier has no secret/public key pair. This can be useful in practical applications.

We first prove that Simp-pkDB^p achieves weak privacy. This helps us to prove Full-pkDB1^H is semi-strong private in SHM.

Theorem 7 (Weak privacy of Simp-pkDB^p). *Assuming the encryption scheme with* $(\mathsf{Enc}', \mathsf{Dec}')$ *is IND-CCA secure and the encryption scheme with* $(\mathsf{Enc}, \mathsf{Dec})$ *is IND-CCA and IK-CPA [3] secure, then Simp-pkDB^p is weak private in PM.*

Proof. $\mathit{\Gamma}_i$ is a game where p_i is the probability that the adversary in $\mathit{\Gamma}_i$ succeeds.

$\mathit{\Gamma}_0$: The adversary $\mathcal{A}$ plays the weak-privacy game in PM. The success probability of $\mathcal{A}$ is p_0.

$\mathit{\Gamma}_1$: We reduce $\mathit{\Gamma}_0$ to $\mathit{\Gamma}_1$ where the verifiers do not decrypt (with Dec') any encryptions sent by the provers and the provers do not decrypt (with Dec) the encryptions generated by the verifiers. Instead, they directly use the values inside the encryption. Because of the correctness of both encryption schemes $p_1 = p_0$.

$\mathit{\Gamma}_2$: We reduce $\mathit{\Gamma}_1$ to $\mathit{\Gamma}_2$ where all provers encrypt (with Enc') a random value instead of pk_P, N and all verifiers encrypt (with Enc) a random value instead of $(C||R||N)$. Note that the change on e_V is indistinguishable by an adversary since it does not know sk_P because we prove here weak privacy. Thanks to the IND-CCA security of the encryption schemes $p_1 - p_2$ is negligible.

$\mathit{\Gamma}_3$: We reduce $\mathit{\Gamma}_2$ to $\mathit{\Gamma}_3$ where the prover does not decrypt (with Dec) the encryptions e_V generated by the adversaries and it aborts. Since N has never been used, the probability that $\mathcal{A}$ sends a valid encryption of N is negligible. Therefore, $p_3 - p_2$ is negligible. Remark that in $\mathit{\Gamma}_3$, $\mathsf{Dec}_{\mathsf{sk}_P}$ has never used.

$\mathit{\Gamma}_4$: We reduce $\mathit{\Gamma}_3$ to $\mathit{\Gamma}_4$ where the prover replaces pk_P by a freshly generated public-key (that $\mathcal{V}$ uses if e_P is correctly forwarded). The only visible change from $\mathit{\Gamma}_3$ is that now e_V is encrypted using a new key. Because of IK-CPA security of the encryption scheme (with $\mathsf{Enc}, \mathsf{Dec}$), $p_4 - p_3$ is negligible.

Now, in $\mathit{\Gamma}_4$, no identity is used by the verifiers and the provers, so adversary succeeds $\mathit{\Gamma}_4$ with $\frac{1}{2}$ probability. Therefore, $p_0 - \frac{1}{2}$ is negligible. □

Simp-pkDB^p is not strong private due to the following attack: Assume that an adversary corrupts a prover P and learns sk_P. Later, he can decrypt all e_V sent by the verifier with sk_P. If e_V is sent to P, then it means the adversary learns the challenges and responses. When these challenges and responses become known during MiM-OTDB, the adversary can identify P.

Theorem 8 (Semi-strong privacy of Full-pkDB1^H). *Assuming that the encryption scheme with* $(\mathsf{Enc}', \mathsf{Dec}')$ *is IND-CCA secure and the encryption scheme with* $(\mathsf{Enc}, \mathsf{Dec})$ *is IND-CCA and IK-CCA [3] secure, then Full-pkDB1^H is semi-strong private in SHM.*

Proof. The proof works like in Theorem 7. We only let non-anonymous hardware decrypt e_V from the adversary with the right key through a CCA query in the IK-CCA game. □

Full-pkDB2^H: Eff-pkDB^p [19] is the most efficient public-key DB protocol which is secure against MiM, DF, DH and strong private. Briefly, in Eff-pkDB^p, after the prover transmits its public key via encryption, V and P run a key agreement protocol with the algorithms $(\mathsf{Gen}_V, \mathsf{Gen}_P, A_{N_V}, B_{N_V}, D)$. In the end, with the agreed key, they run a symmetric DB protocol.

One of the assumptions in MiM-security of Eff-pkDB^p is that the symmetric DB is "one-time multi-verifier MiM-secure"[5] [19]. It is not possible to use

[5] It is equivalent to MiM-security with one prover instance and multiple verifier's instances.

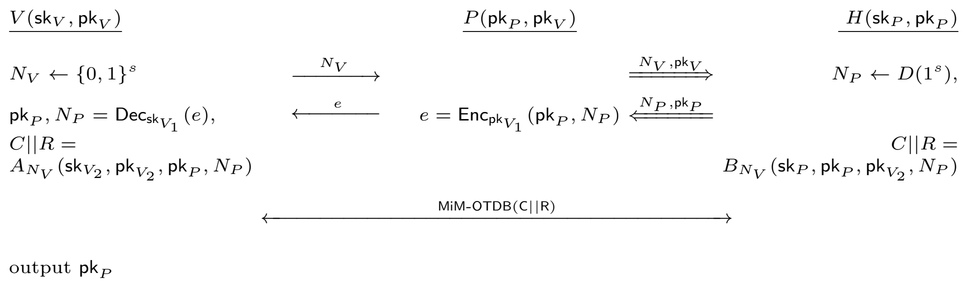
$V(\mathsf{sk}_V, \mathsf{pk}_V)$
$P(\mathsf{pk}_P, \mathsf{pk}_V)$
$H(\mathsf{sk}_P, \mathsf{pk}_P)$
$N_V \leftarrow \{0,1\}^s$
N_V
N_V, pk_V
$N_P \leftarrow D(1^s)$,
$\mathsf{pk}_P, N_P = \mathsf{Dec}_{\mathsf{sk}_{V_1}}(e)$,
e
$e = \mathsf{Enc}_{\mathsf{pk}_{V_1}}(\mathsf{pk}_P, N_P)$
N_P, pk_P
$C||R =$
$A_{N_V}(\mathsf{sk}_{V_2}, \mathsf{pk}_{V_2}, \mathsf{pk}_P, N_P)$
$C||R =$
$B_{N_V}(\mathsf{sk}_P, \mathsf{pk}_P, \mathsf{pk}_{V_2}, N_P)$
MiM-OTDB(C||R)
output pk_P

Fig. 6. Full-pkDB2^H. Double arrow shows the communication with H.

MiM-OTDB on current Eff-pkDB^p as a symmetric DB because MiM-OTDB does not fulfill the assumption. Hence, we modify Eff-pkDB^p so that one time MiM-security is enough. In this way, we are able to use MiM-OTDB as a symmetric DB which does not require any computation.

We slightly change the verifier algorithm of Eff-pkDB^p and convert Eff-pkDB^p into SHM. We call this new version in SHM as Full-pkDB2^H (in Figure 6). The description of Full-pkDB2^H is as follows: The verifier V has the secret/public key pair $(\mathsf{sk}_V, \mathsf{pk}_V) = ((\mathsf{sk}_{V_1}, sk_{V_2}), (\mathsf{pk}_{V_1}, \mathsf{pk}_{V_2}))$ which has two parts where the first part is generated from the key generation algorithm of an encryption scheme and the second part is generated by Gen_V. H has the input $(\mathsf{sk}_P, \mathsf{pk}_P)$ generated by Gen_P. The input of P is $(\mathsf{pk}_P, \mathsf{pk}_V)$. First, V picks a nonce N_V from $\{0,1\}^s$ and sends it to P. Then, P relays it to H. Similarly, H picks N_P from the distribution $D(1^s)$ and gives it P. P encrypts N_P and pk_P with pk_{V_1}. Then, P sends the encryption e to V. V decrypts it with sk_V and learns N_P, pk_P. H and V run the algorithms $B_{N_V}(\mathsf{sk}_P, \mathsf{pk}_P, \mathsf{pk}_{V_2}, N_P)$ and $A_{N_V}(\mathsf{sk}_{V_2}, \mathsf{pk}_{V_2}, \mathsf{pk}_P, N_P)$ which output $C||R$, respectively. In the end, they run MiM-OTDB using $C||R$ as a secret key.

The conversion of Full-pkDB2^H into PM is called as "our variant of Eff-pkDB^p". In this variant, the prover algorithm is $P^{H(\mathsf{sk}_P, \mathsf{pk}_P)}$ where P and H are from Full-pkDB2^H. The difference between the verifier algorithms of Eff-pkDB^p [19] and our variant of Eff-pkDB^p is the following: In Eff-pkDB^p, V does not select any nonce (equivalently, we can say that N_V is a constant) and the algorithms A_{N_V} and B_{N_V} generate a one-time secret key to run a symmetric DB protocol. Remember that we do this change in the verifier algorithm of Eff-pkDB^p to increase its efficiency in SHM since we can use MiM-OTDB with this version.

Theorem 9 (Security of Full-pkDB2^H). *If the key agreement protocol* $(\mathsf{Gen}_V, \mathsf{Gen}_P, A_{N_V}, B_{N_V}, D)$ *is D-AKA secure [19] for all fixed* $N_V \in \{0,1\}^s$ *and MiM-OTDB is* ***one time MiM-secure*** *then Full-pkDB2^H is secure in SHM.*

Proof. We prove it by using Theorem 2. Consider that $DB = (\mathcal{K}_V, \mathcal{K}_P, V, P^H_{dum}, B)$ with V and H from Full-pkDB2^H is MiM-secure in

PM. Actually, DB is our variant of Eff-pkDB. Using Theorem 2, Full-pkDB2^H is secure because our variant of EffpkDB is MiM-secure assuming that the key agreement protocol $(\mathsf{Gen}_V, \mathsf{Gen}_P, A_{N_V}, B_{N_V}, D)$ is D-AKA secure for all fixed $N_V \in \{0,1\}^s$ and MiM-OTDB is **one time MiM-secure**. The MiM security proof of our variant of Eff-pkDB is in the full version of the paper.

$\square$

Full-pkDB2^H achieves almost optimal security bounds because MiM-security of our variant of Eff-pkDB is reduced to MiM-security of MiM-OTDB.

We see that Full-pkDB2^H is secure without encryption. Actually, the encryption is used for achieving privacy. So, if privacy is not a concern, we can use Full-pkDB2^H without the encryption and decryption.

Theorem 10 (Strong privacy of Full-pkDB2^H). *Assuming that the key-agreement protocol* $(\mathsf{Gen}_V, \mathsf{Gen}_P, A_{N_V}, B_{N_V}, D)$ *is D-AKA^p secure [19] for all fixed* $N_V \in \{0,1\}^n$ *and the crypto system is IND-CCA secure, Full-pkDB2^H is strong private in SHM.*

Proof. We first show that our variant of Eff-pkDB^p is strong private in PM. Actually, the strong privacy proof of our variant of Eff-pkDB^p is the same with the proof of Eff-pkDB^p (Theorem 7 of [19]) where first it reduces the privacy game to the game where all the encryptions are random (the reduction showed by using IND-CCA security) and then reduces to the game where the provers use a random secret and public key pair with B_{N_V} (the reduction showed by using D-AKA^p). Because of the equivalence of strong privacy of a DB in SHM and its conversion in PM, we can conclude that Full-pkDB2^H is strong private. $\square$

The prover algorithms of Full-pkDB1^H and Full-pkDB2^H are not P_{dum}, but it can be easily seen from the proofs of Theorem 6 and Theorem 9 that the computations in these algorithms do not have any effect on the security (i.e., the security of Full-pkDB1^H and Full-pkDB1^H do not need any security assumptions on the encryption scheme with $(\mathsf{Enc}', \mathsf{Dec}')$ which is used by P.)

5 Conclusion

In this paper, we defined a new DB with three algorithms and designed its adversarial and communication model of SHM. According to our new model, we define a new security definition. We showed that the trivial attack of TF is preventable in our definition. By showing implications between different threat models, we deduced that if a DB protocol achieves TF-security in SHM, then it is secure against all other security notions. This result cannot be applied in PM because TF-security is not possible. We also gave some security relations between PM and SHM. One of the relations shows that we can construct a DB protocol that is secure against all the threat models including TF in SHM, if its conversion into PM is MiM-secure. This result is significant because it shows that many MiM-secure DB protocols in the literature [5, 24, 4, 6, 17, 19, 26] can be used to achieve higher security level in our model.

We constructed a new only MiM-secure symmetric key DB in PM called MiM-symDB. It achieves optimal security bounds and it is the most efficient DB achieving this. We did not need to achieve other security models with MiM-symDB because MiM-security is enough to have a secure DB protocol in SHM by using P_{dum}. In addition, we constructed another symmetric DB protocol MiM-OTDB. It is MiM-secure when it is run at most one time. It does not require any computation, so it is the most efficient one.

We also considered public key DB protocols in SHM. For this, we derived protocols Full-pkDB1^H and Full-pkDB2^H from Eff-pkDB^p and Simp-pkDB [19], respectively. Full-pkDB1^H was constructed through some modifications on Eff-pkDB^p to be able to use computation free sub-protocol MiM-OTDB. We formally proved that Full-pkDB1^H is strong private and secure in SHM. By adding one extra encryption, we added privacy to Simp-pkDB and constructed Full-pkDB2^H from its private version. We proved Full-pkDB2^H is semi-strong private and secure in SHM.

Compared to the previous models [1, 13, 5] which do not have any practical and secure solution against all the threats, SHM lets us construct more efficient protocols while achieving the **highest security**.

References

1. G. Avoine, M. A. Bingöl, S. Kardaş, C. Lauradoux, and B. Martin. A formal framework for analyzing RFID distance bounding protocols. *Journal of Computer Security–Special Issue on RFID System Security*, 2010.
2. G. Avoine, X. Bultel, S. Gambs, D. Gérault, P. Lafourcade, C. Onete, and J.-M. Robert. A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 800–814. ACM, 2017.
3. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT 2001*, pages 566–582. Springer, 2001.
4. I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Secure and lightweight distance-bounding. In *Lightweight Cryptography for Security and Privacy*, LNCS 8162, pages 97–113. Springer, 2013.
5. I. Boureanu and S. Vaudenay. Optimal proximity proofs. In *Inscrypt*, LNCS 8957, pages 170–190. Springer, 2014.
6. S. Brands and D. Chaum. Distance-bounding protocols (extended abstract). In *EUROCRYPT*, LNCS 765, pages 344–359. Springer-Verlag, 1993.
7. X. Bultel, S. Gambs, D. Gérault, P. Lafourcade, C. Onete, and J.-M. Robert. A prover-anonymous and terrorist-fraud resistant distance-bounding protocol. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 121–133. ACM, 2016.
8. L. Bussard and W. Bagga. Distance-bounding proof of knowledge to avoid real-time attacks. In *Security and Privacy in the Age of Ubiquitous Computing*, IFIP Advances in Information and Communication Technology Volume 181, pages 223–238. Springer, 2005.
9. L. Bussard and Y. Roudier. Embedding distance-bounding protocols within intuitive interactions. In *Security in Pervasive Computing*, pages 143–156. Springer, 2004.

10. S. Capkun, L. Buttyan, and J.-P. Hubaux. Sector: secure tracking of node encounters in multi-hop wireless networks. In *In ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, pages 21–32, 2003.
11. C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 113–127. IEEE, 2012.
12. Y. Desmedt. Major security problems with the "unforgeable" (Feige-) Fiat-Shamir proofs of identity and how to overcome them. In *Congress on Computer and Communication Security and Protection Securicom*, pages 147–159. SEDEP Paris France, 1988.
13. U. Dürholz, M. Fischlin, M. Kasper, and C. Onete. A formal approach to distance-bounding RFID protocols. In *Information Security*, pages 47–62. Springer, 2011.
14. M. Fischlin and C. Onete. Terrorism in distance bounding: modeling terrorist-fraud resistance. In *Applied Cryptography and Network Security*, LNCS 7954, pages 414–431. Springer, 2013.
15. G. P. Hancke. A practical relay attack on iso 14443 proximity cards. *Technical report, University of Cambridge Computer Laboratory*, 59:382–385, 2005.
16. J. Hermans, A. Pashalidis, F. Vercauteren, and B. Preneel. A new RFID privacy model. In *ESORICS*, LNCS 6879, pages 568–587. Springer, 2011.
17. J. Hermans, R. Peeters, and C. Onete. Efficient, secure, private distance bounding without key updates. In *WiSec*, Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, pages 207–218, 2013.
18. H. Kılınç and S. Vaudenay. Optimal proximity proofs revisited. In *ACNS*, pages 478–494. Springer, 2015.
19. H. Kılınç and S. Vaudenay. Efficient public-key distance bounding protocol. In *Asiacrypt*, 2016.
20. C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira. The swiss-knife RFID distance bounding protocol. In *Information Security and Cryptology–ICISC 2008*, pages 98–115. Springer, 2008.
21. J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji. Detecting relay attacks with timing-based protocols. In *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pages 204–213. ACM, 2007.
22. D. Singelee and B. Preneel. Location verification using secure distance bounding protocols. In *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pages 7–pp. IEEE, 2005.
23. D. Singelée and B. Preneel. Distance bounding in noisy environments. In *Security and Privacy in Ad-hoc and Sensor Networks*, LNCS 4572, pages 101–115. Springer, 2007.
24. S. Vaudenay. On modeling terrorist frauds. In *Provable Security*, LNCS 8209, pages 1–20. Springer, 2013.
25. S. Vaudenay. On privacy for RFID. In *Provable Security*, pages 3–20. Springer, 2015.
26. S. Vaudenay. Private and secure public-key distance bounding application to NFC payment. In *FC*, LNCS 8975, 2015.
27. S. Vaudenay. Sound proof of proximity of knowledge. In *Provable Security*, pages 105–126. Springer, 2015.