Private and Secure Public-Key Distance Bounding

Application to NFC Payment — Short Paper

Serge Vaudenay

EPFL
CH-1015 Lausanne, Switzerland
http://lasec.epfl.ch

Abstract. Distance-Bounding is used to defeat relay attacks. For wireless payment systems, the payment terminal is not always online. So, the protocol must rely on a public key for the prover (payer). We propose a generic transformation of a (weakly secure) symmetric distance bounding protocol which has no post-verification into wide-strong-private and secure public-key distance bounding.

1 Introduction

Several wireless payment systems have recently been spread: automatic toll payment systems for vehicles, NFC-equipped credit cards, and NFC-equipped smartphones applications. These methods allow to pay small amounts without any action from the holder other than approaching their device to the payment terminal. I.e., no confirmation is required on the credit card/smartphone and no PIN code or other credential is requested on the payment terminal. The credit card/smartphone is typically passive.

In relay attacks, a man-in-the-middle $\mathcal{A}$ passively relays messages between two participants: a *prover* P and a *verifier* V [9,10]. The prover P is a credit card/smartphone (of the payer) and the verifier V is a payment terminal (of the vendor). $\mathcal{A}$ can be run by two players: a malicious customer $\mathcal{A}_1$ mimicking a payment in a shop to buy some service to V, and a malicious neighbor $\mathcal{A}_2$ to the victim P. $\mathcal{A}_1$ and $\mathcal{A}_2$ relay messages between P and V. The payer (the victim) may not even be aware that their P is activated and communicating. So, relay attacks can be an important threat.

So far, the most promising technique to defeat relay attacks is distance-bounding (DB) [5]. A DB protocol includes several fast challenge/response rounds during which the verifier/vendor sends a challenge bit and expects to receive a response bit within a very short time from the prover/payer. The protocol fails if some response arrives too late or some response is incorrect. Due to the time of flight, if P is too far from V, his time to compute the response is already over when the challenge reaches him.

Here are the traditional threat models for distance-bounding.

- Honest-prover security: *man-in-the-middle attacks* (MiM) (including *impersonation fraud* [1] and the so-called *mafia fraud* [8] which includes *relay attacks*).
- Malicious-prover security: *distance fraud* (DF) [5], in which a malicious prover pretends that he is close although it is not the case; *distance hijacking* (DH) [7], in which the malicious prover relies on *honest* close-by participants; *collusion frauds* (CF) [3] (including the so-called *terrorist fraud* [8]), in which a malicious prover colludes with close-by participants (but without leaking his credentials).

– *Privacy*, where we want that no man-in-the-middle adversary can learn the identity of the prover. Wide/narrow privacy refers to whether the adversary can see if a protocol succeeds on the verifier side. Strong/weak privacy refers to whether the adversary can corrupt provers and get their secret.

For payment systems, we cannot assume an online connection to a trusted server nor a shared secret between the payer and the vendor: we must have a public-key based protocol. We can further wonder which threat models are relevant. Clearly, the man-in-the-middle attacks are the main concern: we must eliminate relay attacks and their possible active extensions. Privacy is also important as payers want to remain anonymous to observers. Since liability is important in payment systems, the payment must be undeniable, so it must be impossible for a malicious payer to claim having not made a payment because he was too far. We also want to be able to catch red handed people who pay with a stolen credit card. So, distance fraud is also a concern.

Table 1. Existing Public-Key Distance Bounding Protocols

protocol	MiM	DF	DH	CF	Privacy	Strong privacy
Brands-Chaum [5]	secure	secure	insecure	insecure	insecure	insecure
DBPK-Log [6]		insecure		insecure	insecure	insecure
HPO [13]	secure	secure		insecure	secure	insecure
GOR [11]	secure	secure	insecure	insecure	insecure	insecure
ProProx [18]	secure	secure		secure	insecure	insecure
privDB	secure	secure	secure	insecure	secure	secure

(Missing entries correspond to absence of proof in either direction.)

So far, only five public-key distance bounding protocols exist: the original protocol by Brands and Chaum [5], the DBPK-Log protocol by Bussard and Bagga [6], the protocol by Hermans, Peeters, and Onete [13] (herein called the HPO protocol), its recent extension by Gambs, Onete, and Robert [11] (the GOR protocol, herein)[1], and ProProx [18] (see Table 1). None of them (except ProProx) resist to collusion frauds. The Brands-Chaum protocol does not resist to distance hijacking [7]. In [2], Bay *et al.* have broken DBPK-Log. Neither the Brands-Chaum protocol nor ProProx protect privacy but the HPO and GOR protocols were designed for this. However, HPO does not offer strong privacy and privacy in GOR can be broken.[2]

We show how to transform a symmetric distance bounding protocol symDB into a public-key distance bounding privDB. We prove that privDB is DF-secure, MiM-secure, DH-secure, and strong-private by assuming some weak form of DF, MiM, and DH security for the underlying symDB and that symDB belongs to the class of protocols with no post-verification. We propose a suitable symDB protocol called OTDB.

[1] The GOR protocol is a bit different from others as it provides *anonymous authentication*. The verifier does not identify the prover in the protocol, but only gets the insurance that he holds a valid key pair. Furthermore, the protocol is deniable: anybody can forge a valid transcript.

[2] This statement will be proven in a subsequent paper.

2 Definitions

We recall and adapt the framework of [4,18]. We assume a multiparty setting in which participants have a *location* and information travels at the speed of light. Participants receive inputs and produce outputs. Honest participants run their purported algorithm. Malicious participants may run an arbitrary probabilistic polynomial-time (PPT) algorithm. The definition below is adapted from [4,18] to accommodate identification protocols and also to bridge public-key and symmetric distance bounding.

Definition 1. *A* distance-bounding protocol (DB) *is a tuple consisting of: 1. PPT* key generation *algorithms K_P and K_V depending on a security parameter λ and producing a key pair $(\mathsf{sk}_P, \mathsf{pk}_P)$ and $(\mathsf{sk}_V, \mathsf{pk}_V)$, respectively; 2. a two-party PPT protocol $(P(\mathsf{sk}_P, \mathsf{pk}_V), V(\mathsf{sk}_V))$, where $P(\mathsf{sk}_P, \mathsf{pk}_V)$ is the* proving algorithm *and $V(\mathsf{sk}_V)$ is the* verifying algorithm*; 3. a distance bound B. At the end of the protocol, $V(\mathsf{sk}_V)$ has a private output and sends a final message Out_V. He accepts ($\mathsf{Out}_V = 1$) or rejects ($\mathsf{Out}_V = 0$). The protocol must be* complete*. I.e., such that "setting up the keys" for (P,V) then making $P(\mathsf{sk}_P, \mathsf{pk}_V)$ and $V(\mathsf{sk}_V)$ interact together, at locations within a distance up to B always makes $V(\mathsf{sk}_V)$ accept ($\mathsf{Out}_V = 1$) and output pk_P.*

For a public-key distance-bounding identification protocol*, "setting up the keys" for P and V means running $K_P \to (\mathsf{sk}_P, \mathsf{pk}_P)$ and $K_V \to (\mathsf{sk}_V, \mathsf{pk}_V)$. For* Symmetric distance bounding protocols*, provers/verifiers are paired and "setting up the keys" for a pair (P,V) means running $K_P \to \mathsf{sk}_P$ then setting $\mathsf{sk}_V = \mathsf{sk}_P$ and $\mathsf{pk}_P = \mathsf{pk}_V = \perp$.*

Moving to noisy settings [16] follows standard techniques which are omitted herein.

Verifiers are assumed to be able to validate pk_P (e.g., by means of a PKI). In what follows, $\mathsf{Validate}(\mathsf{pk}_P)$ denotes this operation.

Security of DB. Like in [4,18], all security notions are formalized by a game with three types of participants: provers, verifiers, and actors. Each participant can have several instances (as they may run several sessions of the protocol at different locations or time). Without loss of generality, actors are malicious. The purported algorithm is P for provers and V for verifiers. There is a distinguished instance of the verifier denoted by $\mathcal{V}$. Instances of participants within a distance to $\mathcal{V}$ up to B are called *close-by*. Others are called *far-away*. We say that the adversary *wins* if $\mathcal{V}$ accepts.

In security models, we only consider without loss of generality one verifier (possibly with several instances) who is further assumed to be honest. In Def. 2–3, we consider without loss of generality only one prover (possibly with several instances) with an identity corresponding to the key pk_P.

Definition 2 ([18]). *We consider the following honest-prover security notion. At the beginning of the game, we set up the keys (following Def. 1) and give pk_V as input to all participants, sk_P as input to the prover instances, and pk_P as input to all malicious participants. The prover is honest. The DB protocol is* MiM-secure *(man-in-the-middle) if for all such settings in which there is no close-by prover, the probability that $\mathcal{V}$ accepts and outputs pk_P is negligible.*[3]

[3] The key generation algorithms accepts as input a security parameter λ which is omitted for simplicity reasons. Hence, $\Pr[\mathcal{V}\ \mathsf{accepts}]$ is a function of λ. We say that $f(\lambda)$ is *negligible* if for every integer d we have $f(\lambda) = O(\lambda^{-d})$ for $\lambda \to +\infty$.

The DB protocol is one-time MiM-secure *if the above is satisfied in settings where there is a single verifier instance and a single prover instance.*

Definition 3 ([18]). *We consider the following malicious-prover security notion. At the beginning of the game, we use an arbitrary PPT algorithm $K(\mathsf{pk}_V)$ instead of K_P in the key setup. The DB protocol is* DF-secure *(distance fraud) if for all such settings where there is no close-by participant except $\mathcal{V}$, the probability that $\mathcal{V}$ accepts and outputs pk_P is negligible.*

Note that the key of the malicious prover is set up maliciously (even depending on pk_V) using an algorithm K which can differ from K_P.

Privacy. So far, the most general and prominent model for privacy is the simulation-based privacy notion in [17] which was enriched in [15]. Hermans et al. [14] presented a simpler privacy model which we call the HPVP model.

Definition 4 (HPVP Privacy [14]). *We consider an adversary playing with the following oracles: 1.* $\mathsf{Create} \to (i, \mathsf{pk}_P)$ *runs K_P and sets pk_P as a valid key for a new prover whose number is i; 2.* $\mathsf{Corrupt}(i) \to \mathsf{state}$ *returns the current state (in permanent memory) of the ith prover; 3.* $\mathsf{Draw}(i, j) \to \mathsf{vtag}$ *draws either the ith prover (if in the left game) or the jth prover (if in the right game) and returns a pseudonym* vtag *(if the prover is already drawn, $\perp$ is returned); 4.* $\mathsf{Free}(\mathsf{vtag})$ *releases* vtag *so that it can be drawn again; 5.* $\mathsf{SendP}(\mathsf{vtag}, m) \to m'$ *sends a message m to a drawn tag* vtag *and gets a response m'; 6.* $\mathsf{Launch} \to k$ *runs a new verifier whose number is k; 7.* $\mathsf{SendV}(k, m) \to m'$ *sends a message m to the kth verifier and gets a response m'; 8.* $\mathsf{Result}(k) \to \mathsf{Out}_V$ *gives the final result (whether the protocol succeeded or not) of the protocol on the kth verifier side. In the privacy game, the adversary interacts with these oracles and guesses if it is left or right. The game is formalized as follows: 1. run $K_V \to (\mathsf{sk}_V, \mathsf{pk}_V)$ and initialize all verifiers with sk_V and all provers and $\mathcal{A}$ with pk_V; 2. pick $b \in \{0,1\}$; 3. let $\mathcal{A}$ interact with the oracles (in the left game for $b = 0$ or the right game for $b = 1$) and make a guess β; 4. $\mathcal{A}$ wins if $\beta = b$. We have privacy if for every PPT adversary $\mathcal{A}$,* $\Pr[\mathcal{A}\ \mathsf{wins}] - \frac{1}{2}$ *is negligible. For* narrow *privacy, the adversary does not use the* Result *oracle. For* weak *privacy, he does not use the* Corrupt *oracle. Otherwise, the adversary is* wide*, respectively* strong.

Distance Hijacking. In distance hijacking [7], the prover is malicious, running an algorithm $\mathcal{A}$. But we add a honest prover $P(\mathsf{sk}_{P'}, \mathsf{pk}_V)$ with another identity P' associated to the key pair $(\mathsf{sk}_{P'}, \mathsf{pk}_{P'})$. The malicious prover runs $\mathcal{A}(\mathsf{sk}_P, \mathsf{pk}_P, \mathsf{pk}_{P'}, \mathsf{pk}_V)$. We formalize distance hijacking for DB protocols consisting of a regular (i.e., time-insensitive) initialization phase, a time-critical challenge phase, and a regular verification phase. $\mathcal{A}$ is playing a man-in-the-middle between $P(\mathsf{sk}_{P'}, \mathsf{pk}_V)$ and $V(\mathsf{sk}_V)$ except during the challenge phase when he remains passive. (See Fig. 1.)

Definition 5. *A DB protocol (K_P, K_V, P, V, B) is* DH-secure *if for all PPT algorithms K and $\mathcal{A}$, the following game makes $\mathcal{V}$ output pk_P with negligible probability:*

1: for public-key DB: $K_P \to (\mathsf{sk}_{P'}, \mathsf{pk}_{P'})$, $K_V \to (\mathsf{sk}_V, \mathsf{pk}_V)$, $K(\mathsf{pk}_{P'}, \mathsf{pk}_V) \to (\mathsf{sk}_P, \mathsf{pk}_P)$; if $\mathsf{pk}_P = \mathsf{pk}_{P'}$, the game aborts
for symmetric DB: $K_P \to \mathsf{sk}_{P'}$, $K \to \mathsf{sk}_P$, set $\mathsf{sk}_V = \mathsf{sk}_P$, $\mathsf{pk}_P = \mathsf{pk}_{P'} = \mathsf{pk}_V = \perp$;

2: *let* $\mathcal{A}$ *run* $\mathcal{A}(\mathsf{sk}_P, \mathsf{pk}_P, \mathsf{pk}_{P'}, \mathsf{pk}_V)$*, let* $\mathcal{V}$*,* $V_1, V_2, \ldots$ *run* $V(\mathsf{sk}_V)$*, let* P'*,* $P'_1, P'_2, \ldots$ *run* $P(\mathsf{sk}_{P'}, \mathsf{pk}_V)$

3: *let* $\mathcal{A}$ *interact with* $P', P'_1, P'_2 \ldots$ *and* $\mathcal{V}, V_1, V_2, \ldots$ *concurrently until the initialization phase ends for* $\mathcal{V}$

4: *let* P' *and* $\mathcal{V}$ *continue interacting with each other until the challenge phase ends for* $\mathcal{V}$*;* $\mathcal{A}$ *receives the exchanged messages but remains passive*

5: *let* $\mathcal{A}$ *continue interacting with* $P', P'_1, P'_2 \ldots$ *and* $\mathcal{V}, V_1, V_2, \ldots$ *concurrently during the verification phase*

A DB protocol is one-time DH-secure *if the above holds when there are no* P'_i *and* V_i*.*

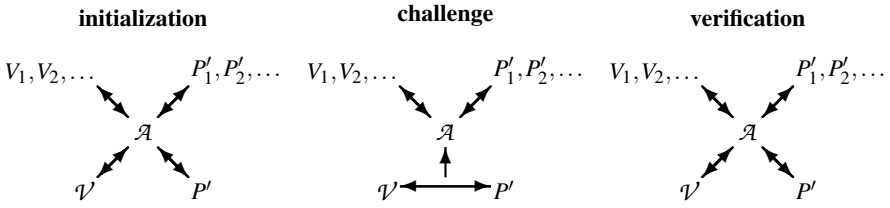

Fig. 1. Distance Hijacking

3 From Symmetric to Asymmetric Distance Bounding

3.1 The OTDB Protocol

We propose a one-time DB protocol OTDB based on the Hancke-Kuhn protocol [12]. It is represented on Fig. 2. We use a $2n$-bit secret s. It is XORed to a random mask m selected by the verifier. The answer to a challenge in iteration i is just the bit of $s \oplus m$ at position $2i-1$ or $2i$, depending on the challenge.

We define a sub-category of simple DB protocols.

Definition 6. *A symmetric DB protocol* (K, P, V, B) *follows the* canonical structure *if there exist 5 PPT algorithms* P_{init}*,* P_{chall}*,* V_{init}*,* V_{chall}*,* V_{ver} *such that* $P(s)$ *and* $V(s)$ *are defined as follows:*

1. $P(s)$ *and* $V(s)$ *run the initialization phase by running* P_{init} *and* V_{init}*, respectively. These algorithms do not use s. They produce a final state* σ_P *and* σ_V*, respectively.*
2. $P(s)$ *and* $V(s)$ *run the challenge phase by running* $P_{\mathsf{chall}}(s, \sigma_P)$ *and* $V_{\mathsf{chall}}(\sigma_V)$*, where* V_{chall} *does not depend on s and produces a final state* σ'_V*.*
3. $V(s)$ *computes* $\mathsf{Out}_V = V_{\mathsf{ver}}(s, \sigma'_V)$*.*

The canonical point is that there is no interactive verification and the secret is only used by P in the challenge phase and by V in the final verification.

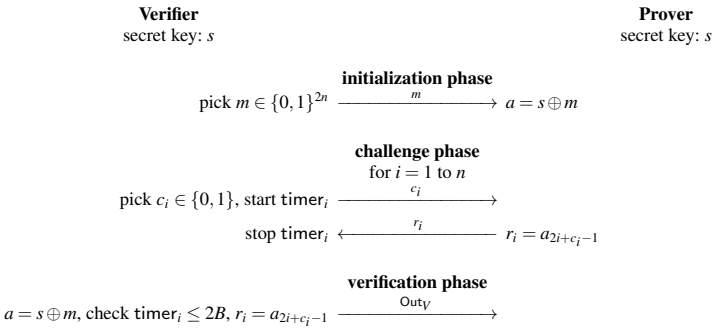

Fig. 2. The OTDB Protocol.

Theorem 7. OTDB *follows the canonical structure. It is DF-secure, one-time MiM-secure, and one-time DH-secure.*

Proof. The canonical structure of OTDB is clear.

For DF-security, we observe that whatever the adversary is doing, the distribution of a on the verifier side is uniform in $\{0,1\}^{2n}$. Since there is no close-by participant, a response can be received on time only if it was sent before the challenge was known. If $a_{2i-1} = a_{2i}$, this can be done with probability 1. Otherwise, this can only be done with probability $\frac{1}{2}$. So, the optimal probability that all responses are correct is $\sum_{w=0}^{n} \binom{n}{w} 2^{-n-w} = \left(\frac{3}{4}\right)^n$ which is negligible.

For one-time MiM-security, we consider a distant $\mathcal{V} = V(s)$ and $P(s)$ with several actors. By playing with $P(s)$, the adversary can deduce for each i either s_{2i-1} or s_{2i} but not both. To answer to $\mathcal{V}$, he must know precisely which of these two bits is needed but when he learns it, it is too late to play with $P(s)$ to get it. So, the probability to pass one round is limited to $\frac{3}{4}$. So, the probability of success is also $\left(\frac{3}{4}\right)^n$, which is negligible.

For one-time DH-security, we consider P' who is set up with a random s and V who is maliciously set up with an independent s'. In the initialization part (which can be corrupted), we let m be the value sent by $\mathcal{V}$ and m' be the value received by P'. When they start the challenge phase, $\mathcal{V}$ uses $a = s' \oplus m$ and P' uses $a' = s \oplus m'$, where m' only depends on m and s'. So, a' is uniformly distributed and independent from a. The challenge part between P' and V cannot be corrupted, by definition of the one-time DH-security. Hence, $\mathcal{V}$ accepts with probability 2^{-n}, which is negligible. □

As concrete parameters, we can use $n = 49$ for a 2^{-20} security. (Since attacks must be online, these levels of security are sufficient in practice.)

3.2 The privDB Protocol

We adapt the RFID protocol from [15,17] for DB. We assume that K_V generates a key pair for a public-key cryptosystem Enc/Dec and that K_P generates a key pair for a digital signature scheme Sign/Verify. The protocol runs as follows (see Fig. 3): 1. V sends a nonce N to P; 2. P picks a random s and sends $\mathsf{Enc}_{\mathsf{pk}_V}(s \| \mathsf{pk}_P \| N \| \mathsf{Sign}_{\mathsf{sk}_P}(N))$ to V; 3. V

decrypts, checks that N is the same, verifies the signature on N, and validates pk_P (if this step fails, V sends $\mathsf{Out}_V = 0$ and aborts); 4. P and V run a symmetric DB symDB based on the secret s (if this step fails, V sends $\mathsf{Out}_V = 0$ and aborts); 5. the private output of V is set to pk_P and the public one is set to $\mathsf{Out}_V = 1$. Compared to HPO [13], the encrypted channel can also be used to transmit a certificate in a private way.

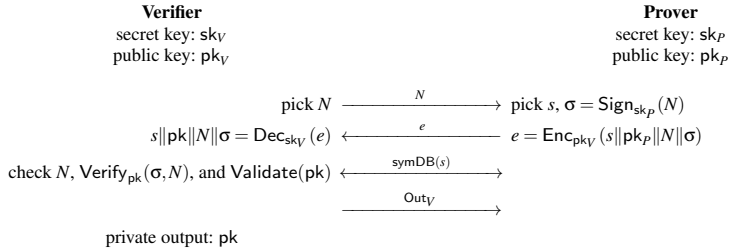

Fig. 3. privDB: Strong Private Public-Key DB from Symmetric DB.

Theorem 8. *If* symDB *is DF-secure then* privDB *is DF-secure.*

The reduction is quite trivial.

Theorem 9. *If 1.* symDB *is one-time MiM-secure, 2. the signature scheme resists to chosen-message universal forgery attacks (UF-CMA secure), 3. the cryptosystem resists chosen-ciphertext attacks (IND-CCA secure), then* privDB *is MiM-secure.*

Proof. We let Γ_0 denote the MiM security game. In what follows, Γ_i is a game and p_i denotes the probability that Γ_i succeeds. We want to show that p_0 is negligible. We first reduce Γ_0 to a game Γ_1 in which no two verifiers select the same nonce. Clearly, $p_1 - p_0$ is negligible. Let N be the value chosen by $\mathcal{V}$. We call the instances of P who are given N the "matching instances". We reduce Γ_1 to a game Γ_2 in which no two matching instances select the same s value. Clearly, $p_2 - p_1$ is negligible. So, two matching instances cannot produce the same e value in Γ_2.

We reduce Γ_2 to a game Γ_3 in which we simulate all verifiers other than $\mathcal{V}$ who receive an e produced by a matching instance. The simulation is made by making these verifiers abort, as this would be the normal behavior due to N mismatch. So, $p_3 = p_2$.

We first consider the case where $\mathcal{V}$ is not given any e coming from a matching instance and show this can only succeed in negligible cases. In this case, no e coming from a matching instance needs to be decrypted. So, we can use the IND-CCA security and replace (with hybrid arguments) each such e by the encryption of a random string and get a similar advantage. We obtain a new game Γ_4 such that $p_4 - p_3$ is negligible. Hence, we can now modify the simulation of matching instances by not computing any signature of N: the signature σ will not be used anyway. We obtain a new game Γ_5 such that $p_5 = p_4$. Now, in Γ_5 we can simulate other prover instances with a signing oracle and use the UF-CMA security with N as the challenge message. Clearly, if $\mathcal{V}$ accepts, there was some forgery. So, due to UF-CMA security, p_5 is negligible.

Now, we are left with the remaining case where $\mathcal{V}$ is given some e coming from a unique matching instance P. We make some further change in the simulation of $\mathcal{V}$ and define Γ_6: we make $\mathcal{V}$ skip the decryption of e and continue with the values encrypted by P. Due to the correctness of the cryptosystem, we have $p_6 = p_3$. Next, we define Γ_7 in which we replace e by the encryption of a random string. Due to IND-CCA security, we have that $p_7 - p_6$ is negligible. We obtain a game Γ_7 in which P selects a random s and runs with $\mathcal{V}$ a single instance of symDB. Note that this s is not used anywhere else, but in symDB, and that s is random. Since P is far away, we are now in the configuration of a one-time MiM attack against symDB. So, we can now invoke the one-time MiM-security of symDB and conclude that p_7 must be negligible. □

Theorem 10. *If* symDB *is one-time MiM-secure, one-time DH-secure, and follows the canonical structure, and if the cryptosystem resists chosen-ciphertext attacks (IND-CCA secure), then* privDB *is DH-secure.*

Proof. We let Γ_0 denote the DH security game. In what follows, Γ_i is a game and p_i denotes the probability that Γ_i succeeds. We want to show that p_0 is negligible.

Since symDB has no interactive verification, Γ_0 consists of two phases after the key set up: the initialization phase and the challenge phase. The last phase matches the challenge phase of symDB between $\mathcal{V}$ and P' alone: we can assume without loss of generality that $\mathcal{A}$, the V_i's and P'_i's play no longer role in this phase. The main point is to realize that for $\mathcal{V}$ to identify pk_P, the $\mathcal{V}$ and P' must start with two independent keys s' and s, and use the one-time DH-security of symDB.

Without loss of generality, we replace $\mathcal{A}$ by the simulation of $\mathcal{A}$ and all P'_i in the DH game. (So, our new adversary is knowing $\mathsf{sk}_{P'}$. But this secret plays no role in DH-security.) So, we can ignore the P'_i's but assume that $\mathcal{A}$ gets $\mathsf{sk}_{P'}$.

We first reduce Γ_0 to a game Γ_1 in which no two verifiers select the same nonce. Clearly, $p_1 - p_0$ is negligible. In what follows, we let N, s, and e denote the values in the protocol as seen by P'. We note that if $\mathcal{V}$ is given e, it will decrypt to $\mathsf{pk}_{P'}$ which is different from pk_P. So, the game cannot succeed. So, we reduce Γ_1 to a game Γ_2 which aborts if e is given to $\mathcal{V}$. We have $p_2 = p_1$.

Next, we reduce Γ_2 to a game Γ_3 in which we simulate as follows the V_i's who receive e. If V_i receives e but did not select N before, we simulate V_i by rejecting the protocol (we know this is his normal behavior in Γ_2). If V_i receives e and did select N before (there can't be several such V_i's), instead of decrypting e, we simulate V_i by directly using the values used by P' during the encryption into e. Clearly, $p_3 = p_2$. We obtain a game in which e is never decrypted.

We reduce Γ_3 into a new game Γ_4 in which the value e is replaced by the encryption of a random string. Since pk_V can be externally set up and that we can outsource all decryption operations in Γ_3 and Γ_4 to a decryption oracle who is never invoked with e, we deduce from the IND-CCA security that $p_4 - p_3$ is negligible.

If no V_i selected N and received e, only P' is set up with a random s to run symDB and no other algorithm receives s in the game. Due to the canonical structure of symDB, the P' run in the initialization phase does not depend on s, so the symDB key by $\mathcal{V}$ is independent from s. So, the challenge phase in Γ_4 is between P' and $\mathcal{V}$ who are set up with independent keys, the one for P' being uniform. By using the one-time DH-security of symDB, this succeeds with negligible probability. So, p_4 is negligible.

If now one V_i selected a and received e (we know that there is no more than one such V_i) we assume this is V_1 without loss of generality. If V_1 does not compute his Out_V before the challenge phase of the game, none of his message depend on s due to the canonical structure of symDB, so we can proceed as in the previous case.

Now, if V_1 sends out his Out_V before the challenge phase of the game, we define a new game Γ_5 in which Out_V is replaced by 0. In Γ_5, we can conclude as in the previous case. So, what is left to be shown is that $\mathsf{Out}_V = 1$ with negligible probability in Γ_4. For that, we observe that P' is only running the initialization of Out_V (so do not depend on s by assumption on Out_V). Since V_1 is set up with a random s and that no other algorithm depends on s during the execution of symDB by V_1, we are in an impersonation attack case. Due to the one-time MiM security of Out_V, we conclude. □

Theorem 11. *If the cryptosystem resists chosen-ciphertext attacks (IND-CCA secure),* privDB *is wide-strong private in the HPVP model.*

Proof. We let Γ_0 denote the wide-strong HPVP privacy game. In what follows, Γ_i is a game and p_i denotes the probability that Γ_i succeeds. We want to show that $p_0 - \frac{1}{2}$ is negligible. We define a game Γ_1 in which we simulate the verifiers who receive some e which was released by any prover by skipping the decryption process and continuing with the correct values which were encrypted by the prover. More precisely, we only take s and N and verify that N is correct. (We know the signature is correct since it was correctly produced by a prover.) Note that the actual value of pk_P which is used as a private output of the verifier is never used in the game. So, only s and N are important for this simulation with a matching e. Due to correctness of the cryptosystem and of the signature scheme, we have $p_1 = p_0$.

Next, we construct Γ_2 in which we replace each e released by a prover by the encryption of a random string using hybrids. Thanks to IND-CCA security, we obtain that $p_2 - p_1$ is negligible.

We observe that in Γ_2, the public keys pk_P and signatures of the prover are never used. So, we define a game Γ_3 in which the simulation of the prover is changed by not computing the signature. Clearly, $p_3 = p_2$.

We now have a game in which provers never use their identity. It does not matter which provers are drawn (the left or the right), the simulation of the prover is the same. So the probability of correctly winning $\beta = b$ must be exactly $p_3 = \frac{1}{2}$. □

4 Conclusion

We have constructed a public-key DB protocol which is the first to be provably DH-secure and the first to be strong private.

References

1. G. Avoine, A. Tchamkerten. An Efficient Distance Bounding RFID Authentication Protocol: Balancing False-Acceptance Rate and Memory Requirement. In *Information Security ISC'09*, Pisa, Italy, Lecture Notes in Computer Science 5735, pp. 250–261, Springer-Verlag, 2009.

2. A. Bay, I. Boureanu, A. Mitrokotsa, I. Spulber, S. Vaudenay. The Bussard-Bagga and Other Distance-Bounding Protocols under Attacks. In *INSCRYPT'12*, Beijing, China, Lecture Notes in Computer Science 7763, pp. 371–391, Springer-Verlag, 2012.
3. I. Boureanu, A. Mitrokotsa, S. Vaudenay. Towards Secure Distance Bounding. In *Fast Software Encryption'13*, Singapore, Lecture Notes in Computer Science 8424, pp. 55–67, Springer-Verlag, 2013.
4. I. Boureanu, S. Vaudenay. Optimal Proximity Proofs. IACR Eprint 2014/693 report, 2014. To appear in the proceedings of INSRYPT'14.
5. S. Brands, D. Chaum. Distance-Bounding Protocols (Extended Abstract). In *Advances in Cryptology EUROCRYPT'93*, Lofthus, Norway, Lecture Notes in Computer Science 765, pp. 344–359, Springer-Verlag, 1994.
6. L. Bussard, W. Bagga. Distance-Bounding Proof of Knowledge to Avoid Real-Time Attacks. In *IFIP TC11 International Conference on Information Security SEC'05*, Chiba, Japan, pp. 223–238, Springer, 2005.
7. C.J. F. Cremers, K.B. Rasmussen, B. Schmidt, S. Čapkun. Distance Hijacking Attacks on Distance Bounding Protocols. In *IEEE Symposium on Security and Privacy S&P'12*, San Francisco, California, USA, pp. 113–127, IEEE Computer Society, 2012.
8. 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'88*, Paris, France, pp. 147–159, SEDEP Paris France, 1988.
9. A. Francillon, B. Danev, S. Čapkun. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. In *Network and Distributed System Security Symposium (NDSS'11)*, San Diego, CA, USA, The Internet Society, 2011.
10. L. Francis, G. Hancke, K. Mayes, K. Markantonakis. On the Security Issues of NFC Enabled Mobile Phones. *International Journal of Internet Technology and Secured Transactions (IJITST)*, vol. 2, pp. 336–356, 2010.
11. S. Gambs, C. Onete, J.-M. Robert. Prover Anonymous and Deniable Distance-Bounding Authentication. In *ACM Symposium on Information, Computer and Communications Security (ASIACCS'14)*, Kyoto, Japan, pp. 501–506, ACM Press, 2014.
12. G.P. Hancke, M.G. Kuhn. An RFID Distance Bounding Protocol. In *Conference on Security and Privacy for Emerging Areas in Communications Networks SecureComm'05*, Athens, Greece, pp. 67–73, IEEE, 2005.
13. J. Hermans, R. Peeters, C. Onete. Efficient, Secure, Private Distance Bounding without Key Updates. In *ACM Conference on Security and Privacy in Wireless and Mobile Networks WISEC'13*, Budapest, Hungary, pp. 195–206, ACM, 2013.
14. J Hermans, A. Pashalidis, F. Vercauteren, B. Preneel. A New RFID Privacy Model. In *Computer Security - ESORICS'11*, Leuven, Belgium, Lecture Notes in Computer Science 6879, pp. 568–587, Springer-Verlag, 2011.
15. K. Ouafi, S. Vaudenay. Strong Privacy for RFID Systems from Plaintext-Aware Encryption. In *Cryptology and Network Security, 8th International Conference CANS'12*, Darmstadt, Germany, Lecture Notes in Computer Science 7712, pp. 247–262, Springer-Verlag, 2012.
16. D. Singelée, B. Preneel. Distance Bounding in Noisy Environments. In *Security and Privacy in Ad-hoc and Sensor Networks ESAS 2007*, Cambridge, UK, Lecture Notes in Computer Science 4572, pp. 101–115, Springer-Verlag, 2007.
17. S. Vaudenay. On Privacy Models for RFID. In *Advances in Cryptology ASIACRYPT'07*, Kuching, Malaysia, Lecture Notes in Computer Science 4833, pp. 68–87, Springer-Verlag, 2007.
18. S. Vaudenay. Proof of Proximity of Knowledge. IACR Eprint 2014/695 report, 2014.