Breaking and Fixing the HB+DB protocol

Ioana Boureanu\textsuperscript{1}, David Gerault\textsuperscript{2}, Pascal Lafourcade\textsuperscript{2}, and Cristina Onete\textsuperscript{3}

\textsuperscript{1} University of Surrey SCCS i.boureanu@surrey.ac.uk
\textsuperscript{2} University Clermont Auvergne LIMOS
david.gerault@uca.fr, pascal.lafourcade@uca.fr
\textsuperscript{3} Université de Rennes 1/IRISA c.onete@gmail.com

Abstract. HB\textsuperscript{+} is a lightweight authentication scheme, which is secure against passive attacks if the Learning Parity with Noise Problem (LPN) is hard. However, HB\textsuperscript{+} is vulnerable to a key-recovery, man-in-the-middle (MiM) attack dubbed GRS. The HB+DB protocol added a distance-bounding dimension to HB\textsuperscript{+}, and was experimentally proven to resist the GRS attack.

We exhibit several security flaws in HB+DB. First, we refine the GRS strategy to induce a different key-recovery MiM attack, not deterred by HB+DB’s distance bounding. Second, we prove HB+DB impractical as a secure distance-bounding (DB) protocol, as its DB security-levels scale poorly compared to other DB protocols. Third, we refute that HB+DB’s security against passive attackers relies on the hardness of LPN; moreover, (erroneously) requiring such hardness lowers HB+DB’s efficiency and security. We also propose a new distance-bounding protocol called BLOG. It retains parts of HB+DB, yet BLOG is provably secure and enjoys better (asymptotical) security.

1 Introduction

In authentication protocols, a prover (e.g., an RFID card) must prove its legitimacy to a verifier (e.g., an RFID reader). The protocol is correct if legitimate provers (almost) always authenticate. Security is expressed in terms of impersonation resistance: a man-in-the-middle (MiM) attacker be able to authenticate with only negligible probability.

The HB protocols. HB and HB\textsuperscript{+} are well-known authentication protocols \cite{11,13}, suitable for resource-constrained devices. HB\textsuperscript{+} is impersonation-secure against passive attackers, if the learning parity with noise (LPN) problem \cite{2} is hard \cite{13}. However, an active MiM can successfully impersonate HB\textsuperscript{+}’s provers, e.g., through GRS attacks \cite{9} (see Section 2.3.).

Distance bounding. Distance-bounding authentication (DB) protocols \cite{7,8} are authentication schemes that moreover aim to deter impersonation attacks mounted via relaying. To this end, the verifier uses a clock to measure the roundtrip time (RTT) during certain exchanges. If these RTTs exceed
a given proximity bound, then the verifier rejects the prover. Besides impersonation security, secure DB must thwart attacks to proximity checking, as described in Section 2.1.

**The HB+DB scheme [15].** HB+DB addressed GRS attacks against HB by adding a proximity-checking countermeasure to that protocol. HB+DB’s effectiveness was assessed by practical experiments, and its limited worst-case security analysis was expressed in terms of LPN-noise levels [15]. In this paper, we present failings of HB+DB, linked both to practice (e.g., DB-security scaling poorly), and to theory (e.g., refute claims that HB+DB’s security against passive attackers relies on LPN-hardness).

**Our contributions.** Our results are as follows.

**DB insecurity.** In Section 3.1, we show how the LPN noise demanded by [16] yields very successful attacks, despite using many proximity-checking rounds.

**Active MiM insecurity.** Section 3.2 exhibits a key-learning, MiM attack against HB+DB, which is GRS-inspired yet more efficient, bypassing HB+DB’s proximity-checking measures.

**Poor Security/Correctness tradeoff.** Section 3.2 shows that the (natural) tradeoff of security and correctness scales poorly for HB+DB: e.g., for a false-rejection rate of 1%, HB+DB yields a 26-bit security for 2048-bit keys.

**LPN-hardness.** Section 3.4 refutes the claim that HB+DB’s passive-security reduces to the hardness of LPN [16].

**Fixing HB+DB.** Using these results, we propose a new DB protocol, BLOG, keeping close to HB+DB. Unlike HB+DB, BLOG attains near-optimal and provable DB security.

**Related work.** Our work, and distance-bounding as a whole, are also related to Secure Neighbourhood Discovery (SND) in ad-hoc networks. SND aims to allow network nodes to establish neighbour tables. However, wormhole attacks (which involve relaying between malicious nodes) prevent SND if no trusted parties or additional, physical location data is known [12]. By contrast, DB protocols take place between only two devices, and a clock is used to verify the prover’s proximity to the verifier. The feasibility of distance-bounding has been demonstrated in [17,10].

2 Preliminaries

2.1 Distance-bounding Security

**Secure distance bounding.** In distance bounding (DB), the prover authenticates not just by correctly responding to challenges by the verifier,
but also by proving that it is within a time/distance bound of $t_{\text{max}}$ from the verifier. Besides impersonation security, DB must also guarantee: 1. Mafia-fraud (MF) resistance: An active MiM attacker cannot authenticate even if given access to an honest prover. 2. Distance-fraud (DF) resistance: A malicious prover located outside the verifier’s proximity cannot successfully authenticate. 3. Terrorist-fraud (TF) resistance: A malicious prover cannot help a MiM attacker authenticate without allowing the adversary to authenticate arbitrarily afterwards. Notably, DB protocols also thwart relay attacks, consisting of the exact forwarding of messages between an honest prover and an honest verifier.

### 2.2 The LPN problem

Let $x$ be a uniformly sampled $k$-bit vector. Let $\eta \in (0, 1/2)$ be a constant noise parameter and $\epsilon$ be an $n$-bit vector with a Hamming weight smaller than $\eta \cdot n$, i.e., $HW(\epsilon) \leq \eta \cdot n$. An instance $LPN_{x,\eta}$ of the LPN problem \cite{LPN} involves solving the equation $r = (A \cdot x) \oplus \epsilon$ in $x$, given a uniformly-sampled $n \times k$ binary matrix $A$ and the $n$-bit vector $r$ produced as shown above. For a matrix $A$ of sub-exponential size (in the security parameter), the best-known algorithms have a sub-exponential time-complexity to solve $LPN_{x,\eta}$ \cite{LPN-alg}.

### 2.3 The HB$^+$ and HB+DB protocols

**The HB$^+$ protocol.** In HB$^+$, the prover $P$ and the verifier $V$ share two $k$-bit long keys $x, y$. In each session, $P$ and $V$ respectively generate two bitstrings $a$ and $b$. $P$ authenticates via several responses $r$, computed as per Figure 1. Each $r$ depends on $x$ and $y$ (blinded by $a$, resp. $b$), and on a noise term $\epsilon$ selected from a Bernoulli distribution with a public mean $\eta \in (1/2)$ (typically, $\eta \in \{1/8, 1/4\}$ \cite{LPN-param}). The round in Figure 1 is repeated $n$ times.

![Fig. 1: One round of the HB$^+$ protocol.](image)

**The security of HB$^+$.** HB$^+$ guarantees only passively-secure authentication. An active MiM attacker can impersonate $V$ and send the all-0 bitstring as $a$ to the prover, thus reducing the HB$^+$ protocol to a one-variable problem (similar to HB \cite{HB}). This MiM can recover $y$ by solving an instance of
The GRS attack. In this attack, the adversary $A$ intercepts the $a$ values from the HB$^+$ verifier, chooses $\delta$ with a Hamming weight 1, and sends $\hat{a} := a \oplus \delta$ to $P$. After forwarding $P$’s responses, $A$ awaits the verifier’s authentication output. If $P$ authenticates despite $A$’s interference, then $A$ concludes that the corresponding bit of $x$ is 0; otherwise, the reverse is assumed to be true. By progressively sending linearly independent $\delta$, $A$ recovers more bits of $x$. Once $x$ is learned, $A$ can impersonate $P$ without knowing $y$, e.g., by successively sending $b=0^k$.

The HB$+$DB protocol. Depicted in Figure 2, the HB$+$DB protocol [15,16] uses two public parameters: $\eta \in (0, 1/2)$ for LPN noise and the mean $\psi \in [0,1/2]$ of a Bernoulli distribution for channel noise. The prover $P$ and the verifier $V$ share three $k$-bit keys $x,y,z$. In the initialisation phase, the prover $P$ chooses a $k$-bit string $s$ and, for $i \in \{1,\ldots,n\}$, $P$ computes: a $k$-bit string $b_i$ (computable in several ways, one of which is $b_i := f_z(s, i)$); an “LPN-noise” bit $\varepsilon_i$ chosen from a Bernoulli distribution with parameter $\eta$; and a bit $c_i := (b_i \bullet y) \oplus \varepsilon_i$. The verifier $V$ computes $b_i$ for the received $s$. In each round of the distance-bounding phase, $V$ starts a clock (added from HB$^+$ into HB$+$DB), then sends a $k$-bit string $a_i$ to $P$. The latter sends a response $r_i := (a_i \bullet x) \oplus c_i$. The verifier receives the response perturbed by channel-noise, $r'_i := r_i \oplus \zeta_i$ (with $\zeta_i$ following a Bernoulli distribution with the mean $\psi$). Upon receipt, $V$ stops the clock and stores the round time $\Delta_i$. During the verification phase, $V$ checks that the $r'_i$’s were within a given toler-

<table>
<thead>
<tr>
<th>Verifier $V$</th>
<th>Prover $P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x \in \mathbb{Z}_2^k, y \in \mathbb{Z}_2^k, z \in {0,1}^k$</td>
<td>$x \in \mathbb{Z}_2^k$</td>
</tr>
<tr>
<td>$f_z : \mathbb{Z}_2^k \times {1,\ldots,n} \rightarrow \mathbb{Z}_2^k$</td>
<td>for $i \in {1,\ldots,n}$: $h_i := f_z(s, i)$</td>
</tr>
<tr>
<td>$\varepsilon_i \in \mathbb{Z}_2$</td>
<td>$a_i \in {0,1}$</td>
</tr>
<tr>
<td>$c_i := (b_i \bullet y) \oplus \varepsilon_i$</td>
<td>$z \in \mathbb{Z}_2^k$</td>
</tr>
</tbody>
</table>

**Fig. 2:** The HB$+$DB protocol.

\[ \sum_{i=1}^n (a_i \bullet x) \oplus \zeta_i \]

where $\zeta_i \in \mathbb{Z}_2$ denotes the channel-noise.
False rejection in HB+DB

The HB+DB protocol [15,16] is essentially a white-box composition of an authentication protocol (i.e., HB+) and a proximity-checking phase. This section shows, however, that HB+DB provides neither secure authentication, nor secure distance bounding. For details, please refer to [4].

3 HB+DB's security shortcomings

The HB+DB protocol [15,16] is essentially a white-box composition of an authentication protocol (i.e., HB+) and a proximity-checking phase. This section shows, however, that HB+DB provides neither secure authentication, nor secure distance bounding. For details, please refer to [4].

3.1 Poor Asymptotic Security in HB+DB

False-acceptance rates in DB are directly proportional to mafia- and distance-fraud resistance [6]. It is thus paramount to tune DB parameters (e.g., number of rounds, noise-levels) to yield tight false-rejection and false-acceptance ratios.

False rejection in HB+DB. The tolerance $\tau$ in HB+DB accounts for the channel and LPN noise, respectively modelled as bits $\zeta_i$ and $\epsilon_i$, following Bernoulli distributions with means $\nu$ and $\eta$. Let $S$ be the random variable described by $\sum_{i=1}^{n}(\epsilon_i \oplus \zeta_i)$. The mean of this variable $S$ is $\mu = n\alpha$, in which $n$ is the total number of time-critical rounds and

$$\alpha := \eta + \nu - 2 \cdot \eta \cdot \nu.$$  \hspace{1cm} (1)

Indeed, an HB+DB response is perturbed iff. exactly one $\epsilon_i$ or $\zeta_i$ values are 1, i.e., with probability $\mathbb{P}[\epsilon = 1] \cdot \mathbb{P}[\zeta = 0] + \mathbb{P}[\epsilon = 0] \cdot \mathbb{P}[\zeta = 1] = \eta \cdot (1 - \nu) + (1 - \eta) \cdot \nu$, which is the value $\alpha$ stated above.

The probability $\mathbb{P}_{FR}^{HB+DB}$ of false rejection in HB+DB accounts for the number of errors lying outside $[\mu - \tau, \mu + \tau]$, namely:

$$\mathbb{P}_{FR}^{HB+DB} = \sum_{i=\mu-\tau}^{\mu+\tau} \binom{n}{i} \cdot \alpha^i \cdot (1 - \alpha)^{n-i}.$$  \hspace{1cm} (2)

False-acceptance rates in HB+DB. Let $r_i^*$ be a random response of an adversary $A$ (a MiM or a malicious prover) for a time-critical round $i$. Let $r_i$ be the response $V$ expects in that round, based on $c_i$. Let the random variable $s_i^*$ be defined as follows: $s_i^* := 1$ if $r_i^* = r_i$ ($A$ was right), and $s_i^* = 0$ if $r_i^* \neq r_i$ ($A$ was wrong). Since the responses $r_i$ are pseudorandom, $\mathbb{P}[s_i^* = 0] = \mathbb{P}[s_i^* = 1] = \frac{1}{2}$. Let $S^*$ be $\sum_i s_i^*$.

The probability $\mathbb{P}_{FA}^{HB+DB}$ that $A$ is falsely accepted, thus committing mafia or distance fraud is:

$$\mathbb{P}_{FA}^{HB+DB} = \sum_{i=0}^{\alpha \cdot n + \tau} \mathbb{P}[S^* = i] = \sum_{i=\alpha \cdot n - \tau}^{\alpha \cdot n + \tau} \binom{n}{i} \left(\frac{1}{2}\right)^i \left(\frac{1}{2}\right)^{n-i}.$$  \hspace{1cm} (2), in which $\tau$ is a fixed ratio of $n$, as prescribed above.
**Tuning HB+DB’s parameters.** In this study, we do the following: 1) we fix $P_{\text{HB+DB}}$ to a reasonable 1% value; 2) we vary the number of rounds and the LPN and channel noise values; 3) using these, we select the tolerance $\tau$ such that $P_{\text{HB+DB}} \approx 1\%$, and then we analyse the false-acceptance probability.

The false acceptance rate (FAR) is the success probability of an adversary committing a MiM attack, MF, or DF by sending (early) random responses; the latter was indeed stated by [15,16] as the best strategy for successful DF against HB+DB. Since HB+DB uses $k$-bit challenges and 1-bit responses, attackers should aim to guess responses, rather than challenges.

Concerning noise, Pagnin et al. [16] specifically require a high $\eta$ value, since “otherwise the LPN-security is lost” c.f. [16], page 11. Typical LPN noise parameters [14] are indeed $\frac{1}{8}$ or $\frac{1}{4}$.

**HB+DB’s asymptotic MF and DF resistance.** Figure 3 shows how the FAR —as given in equation (2)— varies with noise. Each curve represents how FAR probabilities (MF/DF resistance) vary with the number of DB rounds $n$, for a given choice of $\eta$ and $\nu$. In Figure 3, we include $\eta \in \{\frac{1}{4}, \frac{1}{8}\}$ and $\nu \in \{0.05, 0.1\}$.

![Fig. 3: Base 2 log of HB+DB’s false-acceptance rates for $n$ rounds, with $\eta \in \{0.125, 0.25\}$ and $\nu \in \{0.05, 0.1\}$.](image)

Our analysis shows that even for large $n$, HB+DB offers very poor security. Even with a generous 128 rounds, the FAR varies from $2^{-5}$ in the worst case ($\eta=1/4$, $\nu=0.1$) to $2^{-30}$ in the best-case scenario ($\eta=1/8$, $\nu=0.05$).

Figure 4 shows how poorly the FAR scales when increasing the LPN noise. We fixed the channel-noise parameter to a common $\nu=0.05$, and used $n=128$ DB rounds. As $\eta$ approaches $\frac{1}{3}$, the (MF or DF) adversary wins with probability of almost 1, just by sending random responses. In fact, even at $\eta=0.125$, the protocol guarantees only about 30 bits of security.
Comparison with other DB protocols. For a similar requirement of no more than 1\% false rejection rate, and for a channel noise level as high as 0.3, other protocols offer a much higher DF resistance than HB+DB: i.e., 20 bits of security for about 90 rounds for DB1 (q=3) [6] vs. 6 bits of security for the same number of DB rounds in HB+DB.

Causes and impact. As shown, HB+DB has a poor robustness to noise. The three main culprits are: (1) the addition of a non-negligible LPN noise; (2) the verification of a sum of noisy responses rather than of individual values.

Removing LPN-noise from HB+DB. In the case of no LPN-noise, random response-guessing succeeds if the number of errors is below $\tau$, i.e., with the probability of $\sum_{i=0}^{\tau} \mathbb{P}[S^* = i]$.

To keep a false rejection rate below 1\% with $n = 92$ and $\nu = 0.05$, an optimal tolerance $\tau = 10$, i.e., $V$ must tolerate up to 10 errors. For these parameters, the FAR of HB+DB is around $2^{-49}$. Under the same conditions, running DB1 [6] with $q=3$ yields a significantly better FAR (and DF/MF-resistance) of $2^{-90}$. Moreover, DB1 (with $q=3$) is computationally more efficient than HB+DB. Thus, even in the absence of noise, HB+DB compares unfavourably with other DB protocols.

Our take-away message is: using LPN noise as prescribed by [15,16] implies unacceptably-low security levels for HB+DB.

3.2 Key-Learning in HB+DB

Besides being a DB protocol, HB+DB runs all threats of authentication (as it extends HB$^+$), including key-recovery via GRS attacks. Practical experiments by Pagnin et al. indicate that the verifier’s clock detects MiM who try to modify a challenge as in GRS, due to delays in demodulating the
challenge before modifying and re-sending it. We now show how to recover the key without requiring the MiM to demodulate the challenge. Although viewed as somewhat orthogonal to DB, key-recovery attacks do feature in most formal DB models [5,8]. We argue that these attacks are relevant particularly when the recovered key yields an important DB benefit (e.g., accessing a sensitive area); in HB+DB, this is all-the-more important, since recovering just one of its three keys leads to impersonation, i.e., a DB and an authentication vulnerability.

**Our key-recovery attack.** We proceed to describe a concrete MiM attack against HB+DB, featuring an adversary called \( \mathcal{A} \).

Assumptions and setting. The HB+DB scheme uses NRZ-encoded, ASK modulated challenges: each bit is independently encoded into a high- or low-amplitude signal on the carrier link. We make the following additional assumptions: (i) during its attack, \( \mathcal{A} \) can “speak louder” than the verifier, i.e., send a signal with higher power, drowning out (part of) \( V \)’s message, and (ii) \( \mathcal{A} \) knows the time interval between 2 challenges, and the bit period. These are not strong assumptions, if \( \mathcal{A} \) plays a MiM role between \( P \) and \( V \). However, condition (i) might only hold in a probabilistic fashion, if different modulation schemes are used. Additionally, even if the time interval between challenges is unknown to the attacker à priori, \( \mathcal{A} \) can deduce it by observing one or more sessions.

The attacker’s strategy. Instead of reading and modifying challenges, our adversary will just inflict a particular value (e.g., 1) onto one challenge bit, for a given round. To do so, \( \mathcal{A} \) emits a signal stronger than the verifier’s at a specific time. The whole attack is mounted in two steps: (a) key-recovery, when the honest prover is in the verifier’s proximity and the adversary is close to the verifier; (b) impersonation thereafter.

Key-recovery in HB+DB. With \( P, \mathcal{A}, \) and \( V \) positioned as detailed above, \( \mathcal{A} \) now injects a 1 as the bit at position \( j \) in each challenge \( a_i \) received by \( P \) in a given session. Recall that \( \mathcal{A} \) does so not by bit-flipping, but by instantaneously emitting a 1 value “more loudly” at the point corresponding to the \( j \)-th bit-period of challenge \( a_i \). The prover receives a modified \( a_{i,j} \), in which the \( j \)-th bit is replaced by a 1. Then, \( \mathcal{A} \) observes \( V \)’s authentication output, concluding that the \( j \)-th bit of the secret key \( x \) is \( x_{j} = 0 \) if, and only if, the authentication is successful. Else, \( \mathcal{A} \) sets \( x_{j} \) to 1. Repeating this allows \( \mathcal{A} \) to predict the entire key \( x \).

We analyse \( \mathcal{A} \)’s success probability. There are two possibilities for each authentication attempt.

- If \( x_{j} = 1 \), then \( r_i \) will be erroneous in two cases: if \( a_{i,j} \) was originally 0 and was unaffected by LPN or channel noise; or if \( a_{i,j} \) was originally 1, but was affected
by noise. Then the probability that a prover answers wrongly is \( \frac{1}{2} \cdot \zeta + \frac{1}{2} \cdot (1 - \zeta) = \frac{1}{2} \). The session containing an expected \( \frac{n}{2} \), rather than \( \mu \) errors, causing \( V \) to likely refuse the authentication. In this case, \( A \) can deduce that \( x_j = 1 \). In this case, \( P \) is rejected with the same probability that \( n \) Bernoulli trials with mean \( \frac{1}{2} \) yield a number of successes outside the interval \( [n \cdot \alpha - \tau, n \cdot \alpha + \tau] \). Hence, \( A \) guesses the bit of \( x \) correctly with probability of exactly \( 1 - P_{FA} \).

- If the \( j \)-th bit of \( x \) is 0, i.e., \( x_j = 0 \), then injecting a 1 in the challenge at position \( j \) does not alter the expected response. The more formal argument is given in [4].

For random keys, these two scenarios are equiprobable. Hence, the probability to recover one bit of \( x \) is \( P_{active} = \frac{1}{2} \cdot (1 - P_{FR}) + \frac{1}{2} \cdot (1 - P_{FA}) = 1 - \frac{P_{FR} + P_{FA}}{2} \), if \( P_{FR} < \frac{1}{2} \) (if \( P_{FR} \geq \frac{1}{2} \), \( A \) learns less information, since \( V \) sometimes rejects honest provers). Note, however, that a protocol with a correctness lower than \( \frac{1}{2} \) is not practical. Lowering the false acceptance rate, e.g., by using more DB rounds or higher LPN noise, inevitably leads to easier key recovery. This is discussed below.

**Feasibility and HB+DB correctness.** Given its instantaneous bit-changes, our attack bypasses the experimental results of [16,15]. In order to understand the attack’s consequences, we analyse its impact on the (optimal) security parameters.

**Key-recovery vs. FRR.** Our attack allows \( A \) to recover one bit of \( x \) with probability \( P_{active} = 1 - \frac{P_{FR} + P_{FA}}{2} \). Since \( P_{FA} \) and \( P_{active} \) vary in opposite directions, the protocol’s security is optimal when \( P_{FA} = P_{active} \).

![Fig. 5: The lower bound for the best attack probability on HB+DB for \( P_{FR} = 0.01 \) and \( k \) sessions, for each key size \( k \).](image)

Recovering one bit of the key. By solving \( P_{FA} = P_{active} \), we obtain \( 1 - \frac{P_{FR} + P_{FA}}{2} = P_{FA} \iff 2 - (P_{FR} + P_{FA}) = 2 \cdot P_{FA} \iff 3 \cdot P_{FA} = 2 - P_{FR} \iff P_{FA} = \frac{2 - P_{FR}}{3} \).
Thus, regardless of any further parameter-choices, the adversary can either authenticate by sending random responses or recover one bit of the key with a probability \( p \geq 2^{-\frac{2P_{FR}}{3}} \). If \( P_{FR} = 0.01 \), then \( p \geq 0.66 \). Note that this is only a lower bound on the success probability of the best attack, which is independent of the chosen parameters. To lower this probability, one can only increase the false rejection rate (correctness), making the protocol less practical.

Recovering \( x \). Assume the adversary has access to \( k \) sessions, where \( k \) is the size of the key \( x \). The adversary wins if, among these \( k \) sessions, it either:

(i) authenticates at least once with random responses, or
(ii) recovers \( x \).

The probability (i) occurs is \( 1 - (1 - P_{FA})^k \), i.e., 1 minus the chance to fail \( k \) times. The probability (ii) occurs is equal to \((P_{active})^k = (1 - P_{FR} + P_{FA})^k \). The two probabilities increase and decrease with \( p \) respectively, since \( k > 0 \). Where they intersect we have the protocol’s optimal security bound, independently of the choice of parameters. Figure 5 shows this probability for a wide range of key-sizes, from 1 to 2048 bits. Even for a 2048-bit key, HB+DB cannot achieve more than 26 bits of security, while rejecting no more than 1% legitimate authentication attempts. This makes the protocol hardly usable in comparison to other DB protocols, which are faster and provide much better security for much shorter keys.

Quick remedies. Our attack exploits the lack of provable-security analyses against HB+DB. To prevent transcript malleability, transcript-authentication mechanisms can be added at the end of the protocol.

3.3 On the Key-based Security in HB+DB

We now show that recovering \( x \) is sufficient to break both the authentication and distance-bounding properties of the protocol.

Given the key \( x \) output by a key-recovery attacker \( A \), an adversary \( B \) starts a session \( \text{sid} \) with a far-away, honest \( P \), and a separate session \( \text{sid}' \) with \( V \). Its goal is to make \( V \) output 1 at the end of \( \text{sid}' \). In the untimed phases of both \( \text{sid} \) and \( \text{sid}' \), \( B \) just relays \( s \) from \( P \) to \( V \). This is not detected by the verifier’s clock, since it is a not a time-critical exchange. In the fast phase, \( B \) first plays out its session with \( P \), sending all \( a_i \) equal to 0 and getting \( b_i \cdot y + \epsilon_i \) from \( P \). We can safely assume \( B \) uses a noise-cancelling device on this stretch (or the two devices can just be really close). Next, as \( B \) receives valid \( a_i \) values in \( \text{sid}' \) (from \( V \)), it uses \( x \) and the values obtained from session \( \text{sid} \), namely \( b_i \cdot y + \epsilon_i \) for each \( i \), to construct the expected \((a_i \cdot x) \oplus (b_i \cdot y) \oplus \epsilon_i \) responses. Since these are the responses \( V \) expected, \( B \) succeeds with a probability equalling the correctness of the protocol (with respect to the tolerance threshold and the LPN noise).

Quick remedies. One option is to prevent transcript malleability by authenticating the session transcript. Additionally, note that the key \( y \) which
is used in the DB phase adds no extra MiM security; hence, its presence in the protocol is debatable.

### 3.4 HB+DB is Not LPN-based

HB+DB’s very high false-acceptance rates (see Section 3.1) are caused by the protocol’s LPN noise. Pagnin et al., however, require a high noise parameter to achieve security (see p.11 of [16]). Yet, we show that the security of HB+DB cannot be reduced to the hardness of LPN.

**HB+** and LPN. If an HB+ execution [13] were used in the absence of LPN noise (i.e., if $\eta=0$), then a passive adversary against the protocol would be faced with a set of linear equations of the form $a_i \cdot x \oplus b_i \cdot y = r_i$, for publicly-known $a_i, b_i,$ and $r_i$ values. A passive adversary observing $\text{poly}(n)$ sessions can thus solve this system and break security. This is why HB+ requires a non-zero LPN noise $\epsilon$, which ensure that HB+ is as secure against passive adversaries as the hardness of the LPN instance $LPN_{x,\eta}$ used within.

**HB+DB vs. LPN.** Unlike HB+, even a noiseless instance of HB+DB resists passive attacks. This is because a passive adversary against HB+DB is faced with a set of equations of the form $a_i \cdot x \oplus b_i \cdot y = r_i$, but in which only $a_i$ and $r_i$ are public, whilst the $b_i$s remain secret, known only to the honest parties. Thus, $b_i$ randomises the padding to $a_i \cdot x$, which turns an honest execution of HB+DB into a computationally-hard problem for the observing adversary, without it being based on a hard instance of LPN. This aspect is formalised in the long-version of this paper [4].

**Causes and impact.** HB+DB’s security against passive adversaries cannot in fact be reduced to the hardness of LPN since the $b_i$s are only known to the two honest parties, and not observable by the attacker. Despite the lack of proofs for HB+DB in [15,16], HB+DB’s security against passive adversaries is incidentally not lost, relying instead on the pseudorandomness of the function $f$.

### 4 A way to fix HB+DB

We now attempt to fix HB+DB by proposing **BLOG**: a more lightweight and a provably-secure DB scheme.

#### 4.1 The **BLOG** protocol

Next, we show how our design draws from the findings in Sec. 3.

**Removing the LPN noise.** As explained in Section 3.4 HB+DB gains no additional security by using the LPN noise; in fact, the latter is detrimental to HB+DB. So, **BLOG** adds no noise to time-critical responses.
Removing the key $y$. Without the LPN-noise, HB+DB’s response becomes $r_i = a_i \cdot x \oplus b_i \cdot y$. Since each $b_i$ acts as a one-time-pad to $a_i \cdot x$, the responses $r_i$ will have the same, pseudo-random distribution even by omitting $y$, if $b_i$ is drawn at random. We thus save $n \cdot k$ bits of storage and $n$ dot-product computations (in $n$ fast rounds).

Active attacks & the key $z$. In HB+DB, the key $x$ is susceptible to key recovery. Whilst adding a transcript-authentication step could thwart that attack, it makes terrorist-fraud resistance hard to prove. In BŁOG, we follow the recent approach of [1] and replace $x$ by a one-time random string $x_{\text{temp}}$. This will prevent adversaries from using partial leakage of $x_{\text{temp}}$ over multiple sessions. In BŁOG, $x$ is used only to generate fresh $x_{\text{temp}}$ values, similarly to how $z$ used to generate $b_i$ in HB+DB. We also remove HB+DB’s key $z$.

One-bit responses. Following HB+DB, BŁOG also uses $k$-bit challenges and 1-bit responses; yet, BŁOG achieves nearly optimal security bound, in a provable way.

The BŁOG protocol. As depicted in Figure 6, the prover and verifier in BŁOG share only one long-term value $x$. During initialisation, the prover picks a random $k$-bit value $s$, as in HB+DB protocol, and sends it to $V$. Both compute $x_{\text{temp}} || b$ as the output of $f_x(s)$, where $x_{\text{temp}}$ is $k$-bits long and $b$ is $n$-bits long. If $x_{\text{temp}} = 0$ (as indicated by its Hamming weight $HW(x_{\text{temp}})$), $V$ aborts the execution; this abort occurs with a probability of only $\frac{1}{2^k}$, for honest provers. The distance-bounding phase consists of $n$ time-critical exchanges. For each round, $V$ picks a $k$-bit value $a_i$ uniformly at random, starts its clock, and sends $a_i$ to $P$. The prover is expected to reply with
\[ r_i = a_i \cdot x_{\text{temp}} \oplus b_i \text{ (i.e., the inner-product } a_i \cdot x_{\text{temp}} \text{ xor-ed with the } i\text{-th bit of } b) \text{; upon receiving this value, } V \text{ stops its clock and stores the elapsed time } \Delta_i. \]

Finally, in the verification phase, \( V \) checks the correctness of the received \( r_i \), and that \( \Delta_i \leq 2t_{\text{max}} \) for each round, and if so, \( V \) returns an accepting bit. Thus, \( V \) accepts if it holds that \( \forall_{i=1}^{n} ((r_i = a_i \cdot x_{\text{temp}} \oplus b_i) \text{ and } (\Delta_i \leq t_{\text{max}})) \).

### 4.2 The security of BLOG

We now outline BLOG’s security properties, proven in [4].

**Theorem 1.** For the BLOG protocol, if the key \( x \) is chosen uniformly and independently at random and the challenges \( a_i \) are picked independently and uniformly at random by the honest verifier, then the following statements hold.

**DF Resistance.** BLOG is \( q_V \left( \frac{1}{2} \right)^n \)-distant fraud resistant to any adversary opening at most \( q_V \) adversary-verifier sessions.

**MF Resistance.** In addition if \( f \) is a secure PRF, then, for any \((q_{\text{obs}}, q_P, q_V)\)-mafia fraud adversary \( A \) on BLOG, there exists an adversary \( B \) against the security of \( f \) s. that \( \Pr[A \text{ wins}] \leq (q_{\text{obs}} + q_P)^2 \cdot 2^{-k} + \text{Adv}_B^{\text{PRF}} + q_V \cdot \left( \frac{1}{2} + \frac{1}{2^{q_V + 1}} \right)^n 4 + (q_{\text{obs}} + q_P) \cdot 2^{-k} \).

**TF Resistance.** BLOG is \( \text{SimTF} \)-resistant.

### 4.3 An Evaluation of BLOG

**Complexity & security.** BLOG keeps the strong terrorist-fraud resistance of HB+DB’s, while it adds near-optimal mafia- and distance-fraud security. To our knowledge, BLOG is the only distance-bounding protocol to exhibit such strong properties while also guaranteeing provable terrorist-fraud resistance. The DB1 protocol in [6] (for \( q = 3 \)) is the closest to BLOG security-wise but it is computationally more efficient.

We preserve the \( k \)-bit challenge/1-bit response structure of HB+DB. Since \( k \) represents the bit-length of the key, the latter can today be no lower than, say, 80 bits (to prevent trivial brute-force strategies). So, HB+DB and BLOG’s computational complexity cannot be easily lowered, which may mean that their proximity-checks not being as practical.

**Channel-noise in BLOG.** Our protocol’s security is not modulo noisy communications. To make BLOG robust to channel-noise, we would change the verification phase as follows: (1) responses be verified one by one, but only a fraction \( l \) out of \( n \) rounds yield correct responses; (2) all responses be within the time-bound. Then, each bound close to \( 2^{-n} \) attained for DF and MiM-resistance would remain the dominant factor in these resistance-bounds, yet
each would (provably) change to $\text{Tail}(n,m,1-p_{\text{noise}})$, where $\text{Tail}(n,m,1-p_{\text{noise}})$ is the tail of the binomial distribution, i.e., the probability of at least $m$ successes occurring over $n$ trials, and $1-p_{\text{noise}}$ is the chance of one individual success hinging on a response-bit $b$ not being flipped due to the channel-noise. In this noisy case, BLOG’s DF/MF-resistance holds in the DB security models in [8,5]; these proofs are left to an extended version of this paper.

5 Conclusions

In this paper, we have shown that the recently-introduced HB+DB protocol suffers from important problems. In particular, HB+DB does not meet its original goal of thwarting active MiM attacks against HB+. This protocol’s security scales poorly with its correctness, mostly due to unnecessary LPN noise, which was claimed to provide security. Our provable-security analysis of this protocol highlights these flaws and proposes a provably-secure, more efficient version of HB+DB.

Acknowledgements. The authors were supported as follows: I. Boureanu by the EU H2020 Marie Sklodowska-Curie grant 661362 (LvPri20), D. Gerault by the EU FEDER 2014-2020 programme and the regional council of Auvergne, P. Lafourcade by the CNRS PEPS OCA3 project called CHARIOT, and C. Onete by the ANR project 16 CE39 0012 (SafeTLS).

References