

NLHB : A Non-Linear Hopper Blum Protocol

Mukundan Madhavan¹, Andrew Thangaraj¹, Yogesh Sankarasubramaniam², and Kapali Viswanathan²

¹ Indian Institute of Technology, Madras

² HP Labs India, Bangalore

Abstract. In this paper, we propose a light-weight provably-secure authentication protocol called the NLHB protocol, which is a variant of the HB protocol [1]. The HB protocol uses the complexity of decoding linear codes for security against passive attacks. In contrast, security for the NLHB protocol is proved by reducing passive attacks to the problem of decoding a class³ of non-linear codes that are provably hard. We demonstrate that the existing passive attacks([2],[3]) on the HB protocol family, which have contributed to considerable reduction in its effective key-size, are ineffective against the NLHB protocol. From the evidence, we conclude that smaller-key sizes are sufficient for the NLHB protocol to achieve the same level of passive attack security as the HB Protocol. Further, for this choice of parameters, we provide an implementation instance for the NLHB protocol for which the Prover/Verifier complexity is lower than the HB protocol, enabling authentication on very low-cost devices like RFID tags. Finally, in the spirit of the HB⁺ protocol, we extend the NLHB protocol to the NLHB⁺ protocol and prove security against the class of active attacks defined in the DET Model. **Keywords:** HB protocol, LPN problem, Secure and Efficient Authentication Protocol, Passive attacks, RFID tags.

1 Introduction

The HB protocol was proposed in [1] as a low-complexity authentication algorithm that can be computed by human users. Its security is based upon the hardness of the “Learning Parity in Noise” (LPN) problem [4], which is known to be NP-Hard. Though the protocol is secure against passive attacks, where the attacker is allowed only to eavesdrop on protocol communications, it was found to be vulnerable to active attacks, where the attacker could send spurious messages to the protocol participants. Having discovered this efficient active attack against the protocol, Juels and Weis [5] proposed the HB⁺ protocol as an alternative that could resist active attacks. The added complexity of the HB⁺ protocol and the protocol’s need for generation of many random numbers by the Prover rendered it more suitable for low-complexity RFID tags rather than human users.

Cryptanalysis of the HB authentication protocol has resulted in efficient solutions to the LPN problem. Notably, Leviell and Fouque [2] proposed the LF2 algorithm, which is an improved form of the BKW algorithm [6] for solving the LPN problem. Later, Carrijo *et al.* [3] proposed a probabilistic passive attack against HB and HB⁺ protocols. These new solutions have significantly reduced the effective key-size of the HB protocol family that depend on the hardness of decoding linear codes for security against passive adversaries.

In this paper, we define and consider the UNLD problem, which is a decoding problem for a specific class of non-linear codes. We prove hardness of UNLD by reducing the LPN problem to the UNLD problem. Following this, we propose the NLHB protocol, which is a carefully constructed variant of the HB protocol. Security of NLHB against passive attacks is proved by reduction from UNLD to the passive attack problem.

The basic idea behind the NLHB protocol is the use of a carefully-chosen non-linear Boolean function on the linear parities generated in the HB protocol. The use of this non-linear function does not affect the provable security of NLHB as a reduction from the provably-hard UNLD problem still works under a simple uniformity condition satisfied by the function. On the practical side, the use of the non-linear

³ See Section 3 for the definition of the class of non-linear codes.

function considerably weakens the effectiveness of passive attacks like LF2 [2] that depend on the linearity of the parities. Therefore, key efficiency is higher in NLHB when compared to HB.

For implementation, we demonstrate a certain quadratic form chosen from the general family of functions that we propose for the NLHB, which presents a specific low-cost candidate for the protocol. Using this candidate function, the complexity of the NLHB protocol is low enough that it can be implemented in low-cost devices such as RFIDs. Finally, we show that the Prover/Verifier complexity of NLHB protocol can be lower than that of the HB protocol because of the use of smaller keys.

Active attacks similar to those on the HB protocol are possible on the basic NLHB protocol. We demonstrate that the basic NLHB protocol can be extended to an NLHB⁺ protocol, in the spirit of HB⁺, for security in some active attack models. We show that the reductions for the HB⁺ protocol as shown in [7, 8] work for the NLHB⁺ protocol as well.

In summary, the main contribution of this paper is a low-cost, provably-secure extension of the HB protocol through the use of simple non-linear functions on parities. Because of the non-linearity, the proposed NLHB protocol has better resistance to known passive attacks on the HB family resulting in higher key efficiency and cheaper implementations. Also, the NLHB protocol can be modified in the spirit of the several known modifications of the HB protocol to obtain better security against different classes of active attacks.

The paper is organized as follows. In Section 2, we give a brief introduction on the HB and HB⁺ protocols, related security models and the “Learning Parity in Noise” (LPN) problem. In Section 3, we describe the UNLD problem, a type of non-linear code decoding problem and prove its NP-Hardness. This is followed by a description of the NLHB protocol and its security proofs. Section 4 contains discussions on the resistance of the protocol to passive attacks and its Prover complexity. This is followed by the proposition of the NLHB⁺ protocol and its security proofs in Section 5. Section 6 concludes the paper.

2 The HB And HB⁺ Protocols

2.1 HB Protocol

The HB protocol is a symmetric-key authentication protocol. The Prover and Verifier share a random k -bit secret key \mathbf{s} ⁴. The protocol has two public probability parameters $\epsilon, \epsilon' \in]0, \frac{1}{2}[$ such that $\epsilon < \epsilon'$. To authenticate, the Verifier sends a random k -bit challenge vector \mathbf{a} . The Prover, in turn, calculates the binary dot-product $\mathbf{s} \cdot \mathbf{a}$ and replies to the Verifier with $z = \mathbf{s} \cdot \mathbf{a} + v$, where v is a Bernoulli random variable that takes the value 1 with probability ϵ and $+$ denotes XOR addition. This process is repeated n times. At the end of n repetitions, the Verifier returns an “Accept” message iff atmost $\epsilon' n$ responses are “wrong”, i.e, different from dot-products of the secret and the corresponding challenges. This process, which constitutes one authentication session can be parallelized as shown in Figure 1.

In the parallelized form, the Verifier challenges the Prover with a random $k \times n$ matrix, to which

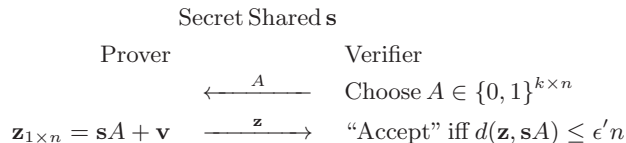


Fig. 1. Parallelized version of the HB protocol

⁴ Refer Appendix D for notations

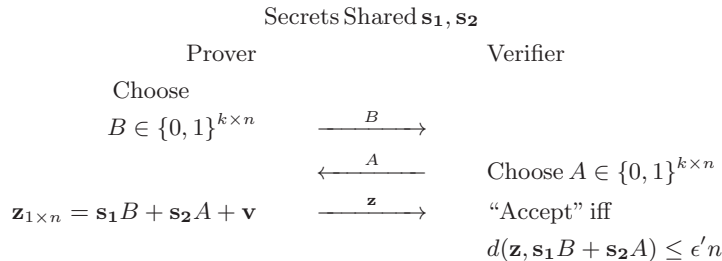


Fig. 2. Parallelized version of the HB^+ protocol

the Prover responds with $\mathbf{z} = \mathbf{s}A + \mathbf{v}$. Here, the bits of the vector \mathbf{v} are all i.i.d Bernoulli random variables with parameter ϵ and the multiplication between the vector \mathbf{s} and A is over the binary field $GF(2)$. The response vector \mathbf{z} is a n -bit vector and the Verifier responds with “Accept” iff $d(\mathbf{z}, \mathbf{s}A) \leq \epsilon' n$, where $d(\cdot)$ denotes Hamming distance. The parameters ϵ, ϵ' , and n are fixed so that both the probability of rejecting an honest Prover as well as the probability of positively authenticating an attacker giving random responses are negligible ([2], Figure 2). The HB Protocol has been proven secure in the Passive attack model as defined below.

Definition 1 (Passive attack model ([5],[7])). *In this model, the adversary algorithm is two-phased. In the first phase (called the query phase), the adversary has access to the transcripts from an arbitrary number of authentication sessions between an honest Prover and Verifier. In the second phase (called the cloning phase), the adversary tries to impersonate an honest Prover to the Verifier.*

However, the HB protocol is not secure against active attacks [5].

2.2 HB^+ Protocol

The HB protocol is susceptible to a simple active attack. In this attack, the attacker repeatedly challenges an honest Prover with the same challenge, and by majority vote over these multiple responses, decides (with high confidence) on the noise-free response. This is repeated for k linearly independent challenges, following which, the secret key is easily found using a Gaussian elimination over the system of linear equations defined by these k challenge-response pairs [5]. Thus, the active attacker need not solve the LPN problem to attack the HB protocol.

To counter such attacks, Juels and Weis [5] proposed the HB^+ protocol (Figure 2). Instead of a single secret, the Prover and Verifier share two k -bit secret keys \mathbf{s}_1 and \mathbf{s}_2 ⁵. In its parallel form, the HB^+ protocol can be described as follows. The Prover starts an authentication session by sending a random “blinding” matrix B to the Verifier, which in turn replies with its own random challenge-matrix A . On receiving A , the Prover responds with $\mathbf{z} = \mathbf{s}_1 B + \mathbf{s}_2 A + \mathbf{v}$. Here, A and B are $k \times n$ matrices, and \mathbf{v} has the same definitions as in the HB protocol. The Verifier responds with an “Accept” decision iff $d(\mathbf{z}, \mathbf{s}_1 B + \mathbf{s}_2 A) \leq \epsilon' n$. Now, when an active attack is mounted, the attacker still has to solve an LPN instance on the matrix B .

The HB^+ protocol is secure against both passive attacks as well as active attacks in a model known as the “DET” attack model.

Definition 2 (DET Attack Model([5],[8])). *In this model, attacks are two-phased. In the first(query) phase, the adversary can interact with an honest Prover an arbitrary number of times. In second (cloning) phase, the adversary interacts with the Verifier and attempts impersonation.*

⁵ The sizes of these secret can be different. This paper shall consider them to be of same size without loss of generality.

Significance Of The “DET” Model: Juels and Weis discuss the significance of the “DET” model in [5], [Appendix A]. Even though this model does not include Man-In-Middle attackers and does not give an attacker access to Verifier decisions at the end of authentication, it is an important security model in a context where the adversary has to forge a valid Prover without the attack being detected. Since attacks in the more powerful prevention-based models like GRS-MIM [9] may not be undetected attacks, the prevention-based model is not the ideal model in all scenarios. As an example, in a setting where the Verifier would report repeated authentication failures from a Prover, the detection-based model is more suitable.

2.3 The LPN Problem and Passive Attacks

Definition 3 (LPN Problem [5]). Let \mathbf{s} be a random binary k -bit vector. Let $\epsilon \in]0, \frac{1}{2}[$ be a constant error parameter. Let A be a random $k \times n$ matrix, and let \mathbf{v} be a random n -bit vector such that $\text{wt}(\mathbf{v}) \leq \epsilon n$, where $\text{wt}(\mathbf{v})$ denotes the Hamming weight of \mathbf{v} . Given A , ϵ and $\mathbf{z} = (\mathbf{s}A) + v$, find a k -bit vector \mathbf{s}' such that $d(\mathbf{z}, \mathbf{s}'A) \leq \epsilon n$.

For large n , this is equivalent to finding the vector \mathbf{s} . The LPN problem has been proven to be NP-Hard [4] and is conjectured to be average-case hard [1]. The BKW algorithm, which was the best-known algorithm to solve the LPN problem when the HB protocol was proposed, has a high complexity and requires a large number of challenge-response pairs $\langle A, \mathbf{z} \rangle$ to obtain a solution. The LF2 algorithm [2], which is an improvement over the BKW algorithm, has considerably lesser complexity and needs lesser challenge-response pairs for its solution. Later, a probabilistic attack on the LPN problem was proposed by Carrijo *et al.* in [3]. These attacks have reduced the effective key-size of the HB protocol, necessitating higher key-sizes. We first describe the LF2 attack, followed by the attack proposed by Carrijo *et al.*

When the key-size is high, exhaustive search over the space of all possible keys is intractable. So, the LF2 attack aims to estimate few bits of the key at a time. The attack involves adding the columns of the challenge-matrix A (and the corresponding responses) so that only the first few (say b) rows have non-zero entries in the resulting matrix. This addition causes two different changes. First, adding two or more noisy responses results in an increased chance of the new response being wrong. So, the apparent Bernoulli parameter in this new set of equations is higher. However, the second and more important change is that, since only the secret bits corresponding to the b non-zero rows play a role in the multiplication $\mathbf{s}A$, the attacker can now find these b bits in isolation by running an exhaustive search over 2^b possibilities. So, by running an exhaustive search over a space of size 2^b (which is much smaller than 2^k), the first b bits of the original key can be found. Repeating this process for the second b rows, and so on, gives the whole key. Thus, the attack depends heavily on the fact that the Prover’s response is a noisy version of some codeword from the linear code having A as its generator.

A second new passive attack was also proposed by Carrijo *et al.* [3]. This attack tries to pick noise-free bits from the response vector and find the key through Gaussian elimination on the system of equations formed from these bits alone. So, this attack too, depends on the Prover’s response being the noisy version of a codeword of the linear code generated by A .

As a consequence of these attacks, a LPN instance using as many as 512 bits of secret can be attacked with a complexity of just 2^{80} operations.

3 The UNLD Problem and the NLHB Protocol

The main idea in this paper is to replace the linear parity generation part sA in the HB protocol with a non-linear version $f(sA)$ for a suitable public function f . The following characteristics are desirable for such a function f :

1. The function f , assumed to be public, must allow for the reduction of hardness from decoding problems to passive attacks on the protocol.

2. The function f must be simple enough to implement on low-cost devices.
3. The function f must provide better resistance to known passive attacks that solve the LPN problem.
4. The function f should allow extensions such as HB^+ for security against active attacks.

We now describe a specific class of non-linear Boolean vector functions and discuss some of its properties that will be used in the security reductions. We discuss the other characteristics like implementation-cost and passive attack resistance in later sections.

3.1 The Function f

Let D and p be positive integers such that $D = n - p$ (n is as described in the HB protocol). We propose the following construction for the NLHB protocol function f . Each bit $y_i; i \in [1, \dots, D = n - p]$ of the output $\mathbf{y} = f(\mathbf{x}); \mathbf{y} \in \{0, 1\}^D, \mathbf{x} \in \{0, 1\}^n$ will be computed as

$$y_i = x_i + g([x_{i+1}, \dots, x_{i+p}]), \quad (1)$$

where x_1, \dots, x_n are the bits of \mathbf{x} and g is a p -bit to 1-bit Boolean function containing strictly non-linear terms. Below, we list some important properties for this class of functions.

1. $f : \{0, 1\}^n \Rightarrow \{0, 1\}^D$
2. f is a non-linear function.
3. For uniformly distributed $\mathbf{x} \in \{0, 1\}^n$, $f(\mathbf{x})$ is uniformly distributed in $\{0, 1\}^D$.

A proof of Property 3 is provided in Appendix C. Intuitively, it can be said that the function $g([x_{i+1}, \dots, x_{i+p}])$ causes the output bits y_i to be non-linearly related to \mathbf{x} and the component x_i helps in balancing the output bit y_i .

As a specific example, the function defined by

$$y_i = x_i + x_{i+1}x_{i+2} + x_{i+2}x_{i+3} + x_{i+3}x_{i+4}, 1 \leq i \leq D \quad (2)$$

is a part of this function family when using $p = 3$. The uniform distribution property for $p = 3$ can be readily verified by exhaustively determining the joint distribution of $\{y_i, y_{i+1}, y_{i+2}, y_{i+3}\}$ for a fixed i . When we set $p = 3$, the function f will take a n -bit vector \mathbf{x} and map it onto a $D = (n - 3)$ bit response vector. As we can see, members of this family like the one described in (2) require very low additional complexity (only 3 AND gates and 3 XOR gates in this case) for implementation and their use in any protocol's implementation will add very little complexity. This can easily be accommodated into any RFID tag, however cheap.

In the next section, we describe how this function family can be used to create a robust protocol. In later sections, we use our specific candidate to demonstrate how their use in the protocol leads to increased passive attack resistance while still maintaining low implementation complexity. However, we would like to point out that our proofs of security hold for all functions in the general class of functions in (1).

3.2 UNLD Problem

Suppose $A_{k \times n}$ is the generator matrix of a linear code. Then all vectors of the form $\mathbf{s}A$ are codewords of this code. When we apply the function f to these codewords $\mathbf{s}A$, i.e, we compute $f(\mathbf{s}A)$, the set of vectors $\{f(\mathbf{s}_i A)\}_{i=1}^{2^k}$ at the output can be viewed as a non-linear code.

We now define the UNLD problem, which (in words) is the problem of decoding the class of non-linear codes defined by f and A as $\{f(\mathbf{s}_i A)\}_{i=1}^{2^k}$.

Definition 4 (UNLD Problem). *Let \mathbf{s} be a random k -bit binary vector. Let $\epsilon \in]0, \frac{1}{2}[$ be a constant error parameter. Let A be a random $k \times n$ binary matrix and let \mathbf{v} be a random D -bit vector such that $\text{wt}(\mathbf{v}) \leq \epsilon D$, where $\text{wt}(\mathbf{v})$ denotes the Hamming weight of \mathbf{v} . Given A, ϵ and $\mathbf{z} = f(\mathbf{s}A) + \mathbf{v}$, find the k -bit vector \mathbf{s} .*

We prove the hardness of the UNLD problem by reducing a random instance of the LPN problem, which is known to be NP-Hard to solve, to the UNLD problem. To show the reduction, we consider an existential algorithm X that can solve the UNLD problem. We construct an algorithm S , which can solve a random LPN instance, when given access to X .

Theorem 1 (LPN reduces to UNLD). *Let A be a random $k \times n$ matrix, \mathbf{v}' be a $(n-p)$ -bit Bernoulli noise vector, and \mathbf{s} be a random k -bit vector. Suppose there exists a probabilistic polynomial-time (PPT) algorithm X with input $\langle A, \mathbf{y} = f(\mathbf{s}A) + \mathbf{v}' \rangle$ that can output \mathbf{s} with probability atleast δ . Then, there also exists a PPT algorithm S that can solve a LPN problem instance $\langle G_{k \times n'}, \mathbf{z} = \mathbf{m}G + \mathbf{v} \rangle$ for randomly chosen \mathbf{m} , Bernoulli noise vector \mathbf{v} and $n' \leq \frac{(n-1)}{p}, k < n'$ with probability at least δ .*

Proof. Let $\mathbf{z} = [z_1, \dots, z_{n'}]$ and $\mathbf{v} = [v_1, \dots, v_{n'}]$ be the constituent bits of the vectors described above. The algorithm S , having access to algorithm X works as follows to solve a random LPN instance $\langle G, \mathbf{z} \rangle$ passed to it.

1. Pick r_i for $1 \leq i \leq n' - 1$ such that $r_i \geq (p-1), \sum_{i=1}^{n'-1} r_i = n - p - n'$.
2. Insert r_i Bernoulli bits between bit z_i and z_{i+1} of \mathbf{z} for $1 \leq i \leq n' - 1$. This gives rise to the vector $\mathbf{y}_{(n-p)} = [z_1, b_1 b_2 \dots b_{r_1}, z_2, b_{r_1+1} \dots b_{r_1+r_2}, z_3, \dots, b_{n-p-n'}, z_{n'}]$.
3. Insert r_i columns of zeros in between columns i and $i+1$ of G ($1 \leq i \leq n' - 1$) to get the matrix A . Insert p columns of zeros after the last column of A . Now, the dimension of A is $k \times n$ and A is of the form $A = [\mathbf{g}_1 \mathbf{0} \mathbf{0} \dots \mathbf{0} \mathbf{g}_2 \mathbf{0} \mathbf{0} \dots \mathbf{0} \dots \mathbf{g}_{n'} \mathbf{0} \mathbf{0} \dots \mathbf{0}]$, where \mathbf{g}_i are the columns of G .
4. Pass $\langle A, \mathbf{y} \rangle$ to X and get back \mathbf{m}' .
5. Return \mathbf{m}' as the estimate of the LPN secret \mathbf{m} .

We now show that S succeeds with probability at least δ . Consider the vector $\bar{\mathbf{x}} = \mathbf{m}A$. We can see that $\bar{\mathbf{x}} = [x_1 \mathbf{0} \mathbf{0} \dots \mathbf{0} x_2 \mathbf{0} \mathbf{0} \dots \mathbf{0} x_3 \mathbf{0} \mathbf{0} \dots \mathbf{0} \dots x_{n'} \mathbf{0} \mathbf{0} \dots \mathbf{0}]$, where $[x_1, x_2, \dots, x_{n'}]$ are the bits of $\mathbf{x} = \mathbf{m}G$. We also see that, since g has only non-linear terms (i.e each term in g is some kind of product of at least two input bits) and $r_i \geq (p-1)$, the vector $f(\bar{\mathbf{x}})$ can be written as $f(\bar{\mathbf{x}}) = [x_1 \mathbf{0} \mathbf{0} \dots \mathbf{0} x_2 \mathbf{0} \mathbf{0} \dots \mathbf{0} x_3 \dots \mathbf{0} x_{n'}]$, as all the product terms from g go to zero.

Let this new vector $f(\bar{\mathbf{x}})$ be called \mathbf{x}' . So, the vector \mathbf{y} is of the form $\mathbf{x}' + \mathbf{v}'$ where $\mathbf{v}' = [v_1, b_1 b_2 \dots b_{r_1}, v_2, b_{r_1+1} \dots b_{r_1+r_2}, v_3, \dots, b_{n-p-n'}, v_{n'}]$ Here, v_i are the Bernoulli bits since they are part of the LPN noise vector \mathbf{v} and b_i are picked to be Bernoulli bits. So, $\mathbf{y} = f(\mathbf{m}A) + \mathbf{v}'$, where \mathbf{v}' is a Bernoulli noise vector. Hence, by definition, X will return \mathbf{m} with probability at least δ . Since S succeeds whenever X succeeds, the probability of success of S is at least δ . \square

We note that it is always possible to pick r_i satisfying the condition in Step 1 for any n and $n' \leq \frac{n-1}{p}$. As an example, one could initially fix $r_i = (p-1); \forall i$. Then $\sum r_i = (n'-1)(p-1) = n'p - p - n' + 1$. Then one can add the difference $(n-p-n') - (n'p - p - n' + 1) = n - n'p - 1$ (which is always positive because of the upper bound on n') to say, r_1 , giving us a new set $\{r_i\}$ satisfying the conditions in Step 1 for any n .

3.3 NLHB Protocol

Having established the hardness of the UNLD problem, we now propose the NLHB protocol that is based on this problem. Figure 3 shows one session of the NLHB protocol. The Prover and Verifier share a k -bit secret \mathbf{s} . The Verifier transmits a random $k \times n$ challenge matrix A to the Prover. On receiving this, the Prover computes $f(\mathbf{s}A)$. Then, it computes $\mathbf{z} = f(\mathbf{s}A) + \mathbf{v}$, where \mathbf{v} is a noise-vector whose bits are all independently distributed according to the Bernoulli distribution with parameter ϵ , just like the noise vector in the HB protocol. Here $\mathbf{s}A$ is a n -bit vector and \mathbf{z} is a D -bit vector. On receiving \mathbf{z} , the Verifier checks whether $d(\mathbf{z}, f(\mathbf{s}A)) \leq \epsilon' D$. If this is true, it returns "Accept". Here too, $\frac{1}{2} > \epsilon' > \epsilon$. Further since the noise-vector is of length D , D has to be large enough (≈ 1000) and (D, ϵ, ϵ') have to satisfy the conditions satisfied by the HB protocol parameters (n, ϵ, ϵ') (see Figure 2 of [2]). For example, $D = 1164$, $\epsilon = .25$ and

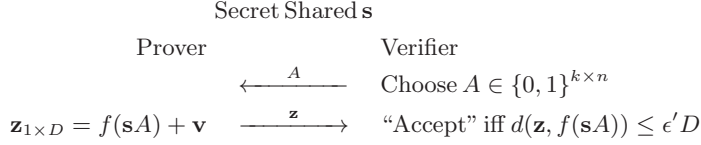


Fig. 3. Parallelized version of the NLHB protocol

$\epsilon' = .348$ is a possible parameter set.

Due to the non-linearity property of f , $f(\mathbf{s}A)$ is some unknown codeword of the random non-linear code $\{f(\mathbf{s}_i A)\}_{i=1}^{2^k}$ and \mathbf{z} is the noisy form of this codeword. To find the secret \mathbf{s} , the attacker now has to decode this random non-linear code instead of the linear code with generator matrix A . We will show in Section 4.1 that existing passive attacks on HB protocol family do not work on our protocol. Our proofs of security are valid for a general class of functions. However, in Section 4, we demonstrate that, for certain choices of f within this family, the protocol complexity is very low.

3.4 Security Proofs For NLHB In Passive Model

The proof of security for NLHB in the passive model involves reductions from the UNLD problem to the forging of the NLHB protocol in the passive model. It is detailed in Theorems 2 and 3. These theorems are broadly based on the proof of security given for the HB protocol in ([7],[8]), with suitable modifications and additions to support the function f . Here, we first prove a technical lemma of our own, that is crucial to the proof of security. We then, use this lemma in the formal proof of our Theorem 2. Since the other parts of the proofs of Theorem 2 and 3 are similar to those in ([7],[8]), we simply give a brief outline here, and delegate the formal proving to the Appendix.

We first explain some brief notations needed for understanding the proof.

Notations In The Proof:

1. The distribution $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ is the distribution followed by the $(kn + D)$ -length bitstrings in the transcript of one authentication session of the NLHB protocol (between honest Prover/Verifier) for a secret \mathbf{s} and error-parameter ϵ . In other words, it is the distribution followed by $\langle A, \mathbf{z} = f(\mathbf{s}A) + \mathbf{v} \rangle$, where A is a random matrix picked from $\{0, 1\}^{k \times n}$ and \mathbf{v} is a Bernoulli noise vector of length D .
2. U_{kn+D} represents uniformly distributed $(kn + D)$ -length bitstrings. In other words, a bitstring S from U_{kn+D} satisfies $\Pr[S = \mathbf{g}] = 2^{-(kn+D)}$; $\forall \mathbf{g} \in \{0, 1\}^{kn+D}$.
3. As already seen in Theorem 1, Algorithm X denotes a UNLD solver algorithm.
4. Algorithm Z is an algorithm that is capable of forging the NLHB protocol in the passive model. Given q bitstrings from the distribution $\mathcal{A}_{\mathbf{s}, \epsilon, f}$, and a challenge matrix A_1 , Z can give a corresponding response \mathbf{z}_1 that will generate an “Accept” response from the NLHB Verifier with non-negligible success probability. In other words, $d(\mathbf{z}_1, f(\mathbf{s}A_1)) \leq u = \epsilon' D$ with non-negligible probability.
5. The Advantage of Z (denoted $Adv_Z^{NLHB-Attack}(k, \epsilon, u, f)$) is defined as the difference between probability of success of Z and the probability of success of an attacker who gives random responses to the Verifier. Since the latter probability is P_{FA} , the probability of false-accept, and is negligible for large D , the advantage of Z is almost the same as its probability of success. The advantage is a function of protocol parameters k, ϵ and u .
6. We also define an intermediate algorithm Y for the purpose of the proof. The algorithm Y is a distinguisher algorithm that can successfully distinguish between the distributions $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ (for secret \mathbf{s}) and

U_{kn+D} . Given q bitstrings from either distribution (which one, is unknown to Y), the algorithm Y processes the bits in some way and outputs 0/1. The probability that Y outputs 1 when the input is $\mathcal{A}_{\mathbf{s},\epsilon,f}$ (for a random \mathbf{s}) and the probability that Y outputs 1 when the input is U_{kn+D} differ significantly. That is,

$$|Pr[\mathbf{s} \leftarrow \{0,1\}^k : Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = 1] - Pr[Y^{U_{kn+D}} = 1]| \geq \delta$$

for some non-negligible probability δ . This difference in probabilities of Y outputting 1 for the different distributions can be used to distinguish these two distributions. In the above equation, $Y^{U_{kn+D}}$ implies that the algorithm Y is inputted bits that follow the distribution U_{kn+D} . The notation $Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}}$ has a similar meaning.

Note that the difference between the probabilities above is for a random key \mathbf{s} . In other words, it is an average over all possible keys.

The reduction is in two steps.

1. In the first step (given in Theorem 2), we prove a reduction from the UNLD problem to the problem of distinguishing between the distributions U_{kn+D} and $\mathcal{A}_{\mathbf{s},\epsilon,f}$, i.e we construct X using Y .
2. In the second step (Theorem 3), we provide a reduction from the problem of distinguishing between these distributions to the problem of forging the NLHB protocol, i.e, we construct Y using Z .

Thus, by using Y as an intermediate, we prove a reduction from the UNLD problem to forging of NLHB protocol.

3.4.1 Theorem 2 Proof Outline

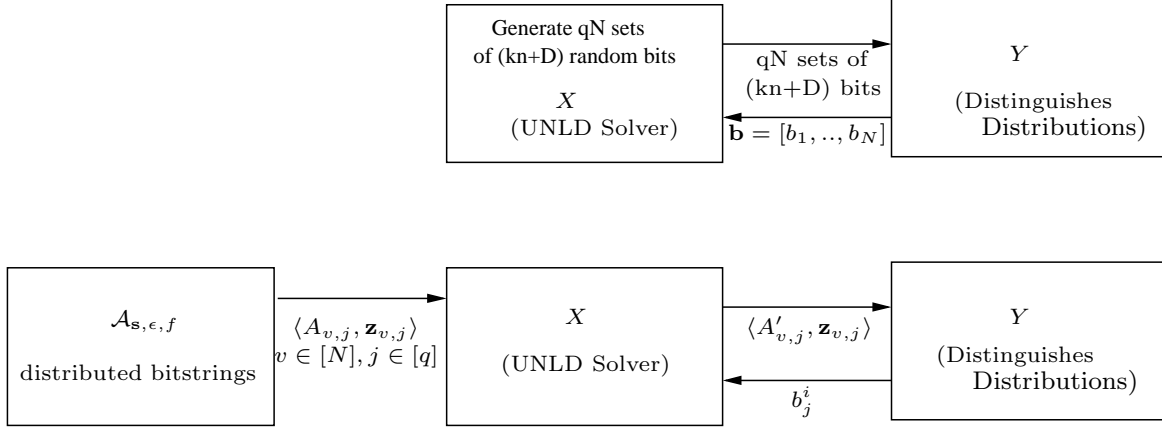
The algorithm X uses Y as follows to solve the UNLD problem. It first estimates p , the probability that Y outputs 1 when given access to U_{kn+D} . To do this, it generates q instances of $(kn + D)$ random bits and passes them to Y (thus simulating U_{kn+D} to Y) and obtains Y 's binary response. By repeating this N times (for reasonably large N) and finding the fraction of 1s in the output, X gets an estimate for p . Next, X takes a bitstring $\langle A, \mathbf{z} \rangle$ from $\mathcal{A}_{\mathbf{s},\epsilon,f}$, to which it has access, by definition. Suppose X wants to find s_i , the i^{th} bit of the secret \mathbf{s} . X adds a random vector \mathbf{c} to the i^{th} row of A . Let us call the resulting matrix A' . Also, let hyb_i denote the distribution followed by the bitstring $\langle A', \mathbf{z} \rangle$. X passes q different instances of hyb_i (for a given i) to Y and obtains its binary output. Like before, it repeats this process N times and estimates p_i , the probability that Y outputs 1 when its input is hyb_i .

If $s_i = 0$, it is easy to see that $hyb_i = \mathcal{A}_{\mathbf{s},\epsilon,f}$. Further, we prove in Lemma 1 below that if $s_i = 1$, then $hyb_i = U_{kn+D}$. Since, by definition, Y outputs 1 with significantly different probabilities for U_{kn+D} and $\mathcal{A}_{\mathbf{s},\epsilon,f}$, p_i will be very close to p if $s_i = 1$ (meaning $hyb_i = U_{kn+D}$) and away from p if $s_i = 0$. So, by estimating the probability of Y outputting 1 and comparing it with p , X can deduce s_i . By repeating this procedure for all $i \in [k]$, X can solve the UNLD problem. We have shown this process in Figure 4. The subscripts of the passed values in the figure denote the qN different instances being sent. We have omitted these subscripts in the above outline for the sake of readability. A more formal treatment is given in the Appendix.

Lemma 1 ($hyb_i = U_{kn+D}$ if $s_i = 1$). *Let A be a randomly chosen $k \times n$ matrix. Let \mathbf{s} be a random k -bit binary secret vector. Further, assign $\mathbf{z} = f(\mathbf{s}A) + \mathbf{v}$, where the bits of \mathbf{v} are i.i.d Bernoulli distributed. Now, let \mathbf{c} be a randomly chosen (independent of all other factors) n -bit binary vector. For an arbitrary $i \in [k]$, let A' denote the matrix formed by modifying only the i^{th} row of A as $(A')_i = (A)_i + \mathbf{c}$. If hyb_i denotes the distribution of the bit-string $\langle A', \mathbf{z} \rangle$, then $hyb_i = U_{kn+D}$ if $s_i = 1$.*

Proof. Consider the conditional probability $Pr[\mathbf{z} = \mathbf{r} \mid A' = \hat{A}]$ for some \hat{A} and an arbitrary $\mathbf{r} \in \{0,1\}^D$.

$$\begin{aligned} Pr[\mathbf{z} = \mathbf{r} \mid A' = \hat{A}] &= Pr[f(\mathbf{s}A) + \mathbf{v} = \mathbf{r} \mid A' = \hat{A}] \\ &= Pr[f(\mathbf{s}A' + \mathbf{c}) + \mathbf{v} = \mathbf{r} \mid A' = \hat{A}] \quad (\text{Since } s_i = 1) \\ &= Pr[f(\mathbf{s}\hat{A} + \mathbf{c}) + \mathbf{v} = \mathbf{r}] \end{aligned}$$



$$p = \text{wt}\left(\frac{\mathbf{b}}{N}\right)$$

$$\mathbf{b}^i = [b_1^i, \dots, b_N^i]$$

$$p_i = \text{wt}\left(\frac{\mathbf{b}^i}{N}\right)$$

$$X \text{ decides } s_i = \begin{cases} 0 & : |p_i - p| \geq \delta/4 \\ 1 & : |p_i - p| \leq \delta/4 \end{cases}$$

Fig. 4. Passing of Strings in the Proof of Theorem 2

We see that since \mathbf{c} is chosen at random, independent of the other variables, $\mathbf{s}\hat{A} + \mathbf{c}$ varies uniformly in $\{0, 1\}^n$. Consequently, $f(\mathbf{s}\hat{A} + \mathbf{c})$ varies uniformly at random in $\{0, 1\}^D$ by Property 3 of f . So, we have

$$\Pr[\mathbf{z} = \mathbf{r} \mid A' = \hat{A}] = \Pr[f(\mathbf{s}\hat{A} + \mathbf{c}) + \mathbf{v} = \mathbf{r}] = 2^{-D} \quad (3)$$

Further,

$$\begin{aligned} \Pr[\mathbf{z} = \mathbf{r}] &= \sum_{\mathbf{x}} \Pr[\mathbf{z} = \mathbf{r} \mid \mathbf{v} = \mathbf{x}] \Pr[\mathbf{v} = \mathbf{x}] \\ &= \sum_{\mathbf{x}} \Pr[f(\mathbf{s}A) = \mathbf{r} + \mathbf{x}] \Pr[\mathbf{v} = \mathbf{x}] \end{aligned}$$

Since A is chosen at random and due to Property 3 of f , we have $\Pr[f(\mathbf{s}A) = \mathbf{r} + \mathbf{x}] = 2^{-D}$. So,

$$\Pr[\mathbf{z} = \mathbf{r}] = \sum_{\mathbf{x}} 2^{-D} \Pr[\mathbf{v} = \mathbf{x}] = 2^{-D} \quad (4)$$

From (3) and (4), we see that \mathbf{z} is independent of A' . So $\Pr[A' = \hat{A}, \mathbf{z} = \mathbf{r}] = \Pr[A' = \hat{A}] \Pr[\mathbf{z} = \mathbf{r}] = 2^{-(kn+D)}$. Since this holds for any arbitrary $\mathbf{r} \in \{0, 1\}^D$, $\text{hyb}_i = U_{kn+D}$ if $s_i = 1$.

Since the function f plays an important role in this lemma, we have presented it here. Since the remaining proof of Theorem 2 is not dependent on the function f , and the proof is adapted from [8, 7], we merely state the theorem here. Please refer to Appendix for detailed proofs.

Theorem 2. (*Reducing UNLD to Distinguishing $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ and U_{kn+D}*): Suppose there exists a probabilistic polynomial-time algorithm Y taking q bitstrings of an unknown distribution (either $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ or U_{kn+D}) and

outputting 0/1, running in time t , such that the probability of outputting 1 when its input is drawn from U_{kn+D} and when its input is drawn from $\mathcal{A}_{\mathbf{s},\epsilon,f}$ differ by at least δ , i.e

$$|Pr[\mathbf{s} \leftarrow \{0,1\}^k : Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = 1] - Pr[Y^{U_{kn+D}} = 1]| \geq \delta.$$

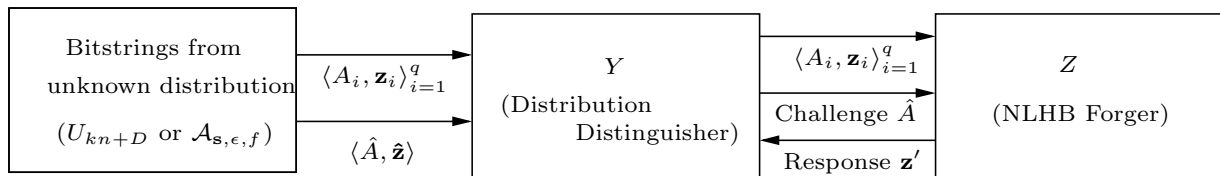
Then there exists X taking $q' = O(q.\delta^{-2}\log(k))$ bitstrings of $\mathcal{A}_{\mathbf{s},\epsilon,f}$ running in time $t' = O(t.k.\delta^{-2}\log(k))$ such that

$$Pr[\mathbf{s} \leftarrow \{0,1\}^k : X^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = \mathbf{s}] \geq \delta/4.$$

Proof. Please refer Appendix A.

3.4.2 Theorem 3 Proof Outline

Having proven the hardness of of the distinguisher problem, we will now reduce it to the passive attack problem, thus proving the hardness of the passive attack problem. The distinguisher algorithm Y can be constructed using algorithm Z as follows. The algorithm Y takes q bitstrings from its unknown distribution, and passes them to Z . In each bitstring, Z treats the first kn bits to be the challenge matrix, and the next D bits to be the corresponding response. This completes the query phase of Z . Now, Y takes one last string $\langle \hat{A}, \hat{\mathbf{z}} \rangle$ from the unknown distribution. It passes \hat{A} , the first kn bits of the string, to Z as a challenge. Let \mathbf{z}' be the response that Z gives Y to this challenge. Then, it can be shown that if the input distribution had been U_{kn+D} , then it is very unlikely that \mathbf{z}' and $\hat{\mathbf{z}}$ are near each other in terms of Hamming distance. On the other hand, if the distribution had been $\mathcal{A}_{\mathbf{s},\epsilon,f}$, then these two are very likely to have Hamming distance below a threshold because of certain properties of the distribution of Z . So, if Y outputs 1 whenever the Hamming distance $d(\mathbf{z}', \hat{\mathbf{z}})$ falls within an appropriately set threshold, the probability of Y outputting 1 for the two distributions will vary significantly, thus fulfilling its requirements. This process is shown in Figure 5. Again, we state only the formal theorem here, and give the complete proof in the Appendix. Using this proof, and the reduction from algorithms X to Y in Theorem 2, we have a reduction from the UNLD problem to the passive attack problem. So, we conclude that the passive attack problem is hard.



Y outputs 1 if $d(\hat{\mathbf{z}}, \mathbf{z}') \leq \epsilon''D$ where ϵ'' is such that $\epsilon' - 2\epsilon'\epsilon + \epsilon < \epsilon'' < \frac{1}{2}$

Fig. 5. Passing of Strings in the Proof of Theorem 3.

Theorem 3. (*Reduction From Distinguishing $\mathcal{A}_{\mathbf{s},\epsilon,f}$ and U_{kn+D} To Forging NLHB Protocol in Passive Model*): If $Adv_Z^{NLHB-attack}(k, \epsilon, u, f) = \delta$ is non-negligible for some polynomial time adversary Z , then the UNLD problem can be efficiently solved.

Proof. Refer Appendix A.

4 Implementation and Efficiency

In this section, we consider the specific low-cost candidate for f given in (2) and demonstrate how existing passive attacks on the HB protocol fail against the NLHB protocol. Then, we compare the Prover complexity of NLHB and HB protocols and demonstrate that the NLHB Prover is required to carry out lesser operations when compared to a HB prover that achieves the same level of security.

4.1 Resistance Against Current Passive Attacks

Using the specific f in (2), we will show how the existing LF2 attack on LPN is ineffective on the NLHB protocol. Let $\mathbf{x} = [x_1, \dots, x_n] = \mathbf{s}A = [\mathbf{s} \cdot \mathbf{a}_1, \dots, \mathbf{s} \cdot \mathbf{a}_n]$, where $[\mathbf{a}_1, \dots, \mathbf{a}_n]$ are columns of A . Let $\mathbf{y} = f(\mathbf{x})$. Then, the passive adversary to NLHB has access to $\mathbf{z} = \mathbf{y} + \mathbf{v}$.

As explained in Section 2.3, the LF2 (or BKW) algorithm works by repeatedly adding the columns of the matrix A and obtaining the response corresponding to this new matrix by adding the responses corresponding to the added columns. We examine the result when the attacker does one column addition. Let the attacker modify A into $A' = [\mathbf{a}_1, \dots, \mathbf{a}_j + \mathbf{a}_k, \dots, \mathbf{a}_n]$, i.e, he adds the k^{th} column to the j^{th} column. The corresponding matrix product between \mathbf{s} and A' will be $\bar{\mathbf{x}} = [x_1, x_2, \dots, x_j + x_k, \dots, x_n]$, i.e $\bar{\mathbf{x}}$ has the same bits as \mathbf{x} except at the j^{th} position, where it is $x_j + x_k$. Let $\bar{\mathbf{y}} = f(\bar{\mathbf{x}})$. Now let us compare the relation between the unnoised responses \mathbf{y} and $\bar{\mathbf{y}}$. As can be seen, the only output bits getting affected by the change of matrix are the ones with indices $(j-3), (j-2), (j-1), j$. We readily see the following relationships.

$$\begin{aligned}
y_{j-3} &= x_{j-3} + x_{j-2}x_{j-1} + x_{j-1}x_j + x_jx_{j-2}. \\
y_{j-2} &= x_{j-2} + x_{j-1}x_j + x_jx_{j+1} + x_{j+1}x_{j-1}. \\
y_{j-1} &= x_{j-1} + x_jx_{j+1} + x_{j+1}x_{j+2} + x_{j+2}x_j \\
y_j &= x_j + x_{j+1}x_{j+2} + x_{j+2}x_{j+3} + x_{j+3}x_{j+1}. \\
\bar{y}_{j-3} &= x_{j-3} + x_{j-2}x_{j-1} + x_{j-1}(x_j + x_k) + (x_j + x_k)x_{j-2}. \\
\bar{y}_{j-2} &= x_{j-2} + x_{j-1}(x_j + x_k) + (x_j + x_k)x_{j+1} + x_{j+1}x_{j-1}. \\
\bar{y}_{j-1} &= x_{j-1} + (x_j + x_k)x_{j+1} + x_{j+1}x_{j+2} + x_{j+2}(x_j + x_k). \\
\bar{y}_j &= x_j + x_k + x_{j+1}x_{j+2} + x_{j+2}x_{j+3} + x_{j+3}x_{j+1}.
\end{aligned}$$

Let us denote the errors between these corresponding bits as $E_{j-3}, E_{j-2}, E_{j-1}, E_j$. From the above equations, we get

$$\begin{aligned}
E_{j-3} &= y_{j-3} + \bar{y}_{j-3} = x_{j-1}x_k + x_kx_{j-2}, \\
E_{j-2} &= x_{j-1}x_k + x_kx_{j+1}, \\
E_{j-1} &= x_{j+1}x_k + x_kx_{j+2}, \\
E_j &= x_k.
\end{aligned}$$

Each error term above is an unknown bit to the attacker, since he does not have access to either a noised or un-noised version of these terms. So, the attacker has to guess the error bits $E_{j-3}, E_{j-2}, E_{j-1}, E_j$ that need to be added to the new response to get the estimate of responses corresponding to the new matrix. The amount of uncertainty involved in guessing these bits can be found from the entropy of $[E_{j-3}, E_{j-2}, E_{j-1}, E_j]$. Since the bits x_i are uniformly distributed, it can easily be seen that this entropy is equal to 2.5 bits. So each time a column is added, the attacker has to guess 2.5 bits on an average. Since there are many such additions needed in the LF2 attack, this attack is no longer feasible against the NLHB protocol. In Table 1, we give the values of the entropy of the bit-wise error terms for different choices of $p = 2, 3, 4$ and functions. For $p = 4$, we have shown only few functions out of the many that achieve the

p	Function Achieving Maximum entropy for given p	Maximum Entropy Achieved for given p
2	$y_i = x_i + x_{i+1}x_{i+2}$	2
3	$y_i = x_i + x_{i+1}x_{i+2} + x_{i+1}x_{i+3}$	2.5
	$y_i = x_i + x_{i+1}x_{i+3} + x_{i+2}x_{i+3}$	2.5
	$y_i = x_i + x_{i+1}x_{i+2} + x_{i+2}x_{i+3} + x_{i+3}x_{i+1}$	2.5
4	$y_i = x_i + x_{i+1}x_{i+4} + x_{i+2}x_{i+3}$	3
	$y_i = x_i + x_{i+1}x_{i+4} + x_{i+2}x_{i+4} + x_{i+3}x_{i+4}$	3
	$y_i = x_i + x_{i+1}x_{i+4} + x_{i+2}x_{i+3} + x_{i+3}x_{i+4}$	3

Table 1. Maximum Entropy Achieved Over All Functions For A Given p and The Function Achieving This Maximum

maximum entropy of 3. As we can see, the entropy increases with increase in p , meaning that LF2 attacks are harder for higher p .

Similar arguments can be given for the infeasibility of the Imai [3] attack, that also relies heavily on linearity. The Imai attack attempts to isolate bits of the response vector that are noise-free and process them to obtain the secret key through Gaussian elimination. However, due to the nonlinear nature of f , Gaussian elimination is not possible with NLHB. Instead, the attacker must solve around a system of $(k + \gamma)$ nonlinear equations in k variables (one for each bit of the secret key). Considering that the number of variables is large, it would be interesting to see if such an attack that involves repeatedly solving systems of nonlinear equations can be efficiently mounted.

The infeasibility of passive attacks on the related HB protocol indicates that the NLHB protocol can achieve 80-bit security using key sizes smaller than 512 bits, which is the number of key bits needed by the HB protocol. Added to this, the fact that no passive solutions exist to the problem of decoding the random non-linear codes described here (and for decoding of random non-linear codes in general) implies that it is reasonable to use key sizes very close to 80 bits with this protocol. However, as a safe value for the key-size, we suggest using 128-bit keys as secrets to resist all known passive attacks on the HB protocol.

4.2 Comparison of Prover Complexity of NLHB and HB

Since each scalar multiplication in the binary field requires one AND gate and one binary addition requires one XOR gate, we calculate the Prover (or Verifier) algorithm’s complexity in terms of binary additions and scalar multiplications. Further, since the complexity involved in adding noise is the same in both protocols, we compare the complexity involved in the calculation of the un-noised responses in the Prover (or Verifier).

The response calculated by the HB protocol for a given random matrix challenge $A_{k \times n}$ is given by $\mathbf{z} = \mathbf{s}A + \mathbf{v}$. The matrix product $\mathbf{s}A$ requires kn scalar multiplications and $(k - 1)n$ (binary) additions for its calculation. Assuming that $\epsilon = .25$ and $\epsilon' = .348$, the length of the final vector to which noise is added should be $n = 1164$ [2]. The value of k for the HB protocol to achieve 80-bit security is around $k = 512$.

In the NLHB protocol, we have a $k \times (D + p)$ challenge-matrix A which we use to find $\mathbf{s}A$. This requires $k(D + p)$ scalar multiplications and $(k - 1)(D + p)$ additions. Further, for the NLHB protocol, we have to evaluate the function f over this vector. If we assume that we use the function f in (2) (with $p = 3$), we require $3D$ scalar multiplications and $3D$ additions for evaluating the function f . So to calculate $f(\mathbf{s}A)$, we need $k(D + 3) + 3D = kD + 3k + 3D$ multiplications and $3D + (k - 1)(D + 3) = kD + 2D + 3k - 3$ additions. Since we add a length- D noise vector in NLHB, D has to be 1164.

For the sake of comparing complexities, if we assume a high-security version of NLHB which uses $k = 512$, then we see that NLHB needs a total of 600996 multiplications and 599829 additions, whereas HB protocol requires 595968 multiplications and 594804 additions. This approximately represents a 0.85% increase in the both the number of multiplications and additions. This shows us that even in comparison

	k	ϵ	ϵ'	Size of Challenge Matrix	Length Of Prover Response	Scalar Multiplications	Scalar Additions
HB	512	.25	.348	512×1164	$n=1164$	595968	594804
NLHB	128	.25	.348	128×1167	$D=1164$	152868	151701

Table 2. Comparison of Prover/Verifier Complexities between NLHB and HB for f with $p = 3$, False-Reject Probability $P_{FR} = 2^{-40}$ and False-Accept Probability $P_{FA} = 2^{-80}$ and 80-bit security.

to a HB protocol using the same keysize as the NLHB protocol, the addition in complexity due to the introduction of f is very small.

However, with $k = 128$, the computation of noise-free NLHB response requires 152868 scalar multiplications and 151701 additions, which is far less than the number of computations needed for a HB protocol Prover to achieve the same level of security, which requires about 512 secret bits.

5 NLHB⁺ Protocol : Extending NLHB To Achieve Security in “DET” Model

Though the NLHB protocol is secure against a passive adversary, it is not secure against an active attacker. An efficient active attack similar to the one demonstrated against HB can also be mounted on the NLHB protocol. So, in the spirit of the HB⁺ protocol, we propose the NLHB⁺ protocol to provide complete security in the DET model.

Figure 6 shows the NLHB⁺ protocol. The Prover and Verifier share two secrets \mathbf{s}_1 and \mathbf{s}_2 . Here,

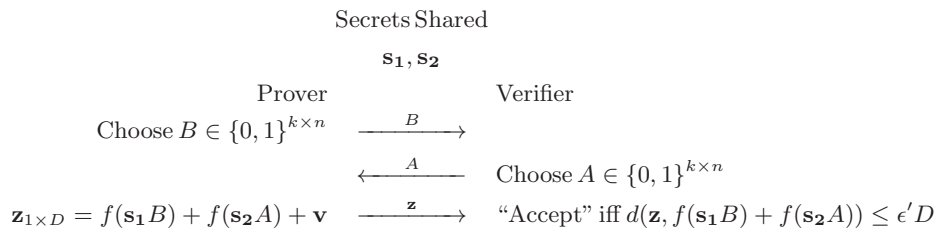


Fig. 6. Parallelized version of the NLHB⁺ protocol

the authentication session is started when the Prover transmits a random $k \times n$ blinding matrix B to the Verifier, which responds with a random $k \times n$ challenge matrix A . The Prover responds with $\mathbf{z} = f(\mathbf{s}_1 B) + f(\mathbf{s}_2 A) + \mathbf{v}$, where f and \mathbf{v} are as defined in the NLHB protocol. The Verifier replies with “Accept” iff $d(\mathbf{z}, f(\mathbf{s}_1 B) + f(\mathbf{s}_2 A)) \leq \epsilon' D$. NLHB⁺ depends on the hardness of the UNLD problem for security against passive attacks. In addition, NLHB⁺ is secure against active attacks in the “DET” model as shown in the next section.

5.1 Security Proof for NLHB⁺ In the “DET” Model

The security proof for NLHB⁺ in the “DET” model is given in Theorem 4, which gives a reduction to active attacks on the NLHB⁺ protocol from the problem of differentiating $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ and U_{kn+D} . Since the latter problem has already been proven hard, this proves the hardness of active attacks. The strategy for Theorem 4 is broadly based on the proofs given in [8] with appropriate modifications to accommodate the function f . So, we simply give an outline of the theorem and the theorem statement here. Please refer the appendix for the complete proof.

First, we give some relevant definitions.

1. The algorithm Z_+ is a polynomial-time NLHB⁺ active adversary. It is a two-phased algorithm. In its query phase, it takes a $k \times n$ random matrix B as input. It then responds with a challenge matrix A (which can be non-random). It should then be given the response that would be given by a legitimate NLHB⁺ Prover for this B and A , i.e., $\mathbf{z} = f(\mathbf{s}_1 B) + f(\mathbf{s}_2 A) + \mathbf{v}$ for secrets \mathbf{s}_1 and \mathbf{s}_2 . In its challenge phase, Z_+ first sends a random blinding matrix \hat{B} to the NLHB⁺ verifier. It then receives a challenge matrix \hat{A} from the Verifier and generates response $\hat{\mathbf{z}}$ that can generate "Accept" from the NLHB⁺ Verifier with non-negligible probability.
2. $Adv_{Z_+}^{\text{NLHB}^+ \text{ attack}}(k, \epsilon, u, f)$ denotes the advantage for an active adversary Z_+ to the NLHB⁺ protocol. It is defined as the difference in probabilities of success of Z_+ and a random attacker. Since the latter is P_{FA} , the probability of false-accept, and is negligible for large D , the advantage is almost the same as the probability of success of Z_+ . The advantage is a function of the parameters k , ϵ and u .

5.1.1 Outline of Proof of Theorem 4

The goal is to construct an algorithm Y that can differentiate U_{kn+D} from $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$, which is the NLHB distribution with secret \mathbf{s}_1 . To simulate a NLHB⁺ Prover with two secrets to the algorithm Z_+ , Y generates a random vector \mathbf{s}_2 to be used as the second NLHB⁺ secret. Now, Y obtains the $(kn + D)$ -length bitstring from the unknown distribution. We denote the first kn bits of this string as \overline{B} and the last D bits as $\overline{\mathbf{z}}$. Y passes \overline{B} to Z_+ , which responds with a challenge matrix A . Now, Y responds with $\mathbf{z} = \overline{\mathbf{z}} + f(\mathbf{s}_2 A)$. Note that if the input distribution had been $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$, this is exactly the response expected by Z_+ for the secret pair $(\mathbf{s}_1, \mathbf{s}_2)$. This process is repeated q times to complete the query phase of Z_+ .

In the challenge phase, the main trick used by Y is that of rewinding Z_+ . After receiving a blinding matrix B from Z_+ , it sends a matrix $A^{(1)}$ and receives response $\mathbf{z}^{(1)}$ from Z_+ . Now, it rewinds Z_+ to the point where it sent B , and sends another challenge $A^{(2)}$ and receives $\mathbf{z}^{(2)}$ for the same B . By summing these responses $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$, the effect of the unknown \mathbf{s}_1 can be removed. This is because $\mathbf{z}^\oplus = \mathbf{z}^{(1)} + \mathbf{z}^{(2)}$ is simply a noisy version of $\hat{\mathbf{z}} = f(\mathbf{s}_2 A^{(1)}) + f(\mathbf{s}_2 A^{(2)})$. Now it is easy to make statements about the distance between these two vectors. It can be shown that in case the distribution is U_{kn+D} , the probability that these vectors are "close" (within a threshold distance) is low, and that if the distribution is $\mathcal{A}_{\mathbf{s}, \epsilon, f}$, then this probability is a non-negligible function of δ (the advantage of Z_+), which is assumed to be non-negligible. So, Y is able to output 1 with very different probabilities for the two distributions, thus helping us differentiate them. The passing of strings in this algorithm construction is shown in Figure 7. Since we already know that UNLD reduces to the problem of differentiating these distributions, we can now say that UNLD reduces to the active attack problem. So, the active attack problem is hard. We now state the formal theorem and give the complete proof in the appendices.

Theorem 4. *If for some polynomial-time adversary Z_+ , $Adv_{Z_+}^{\text{NLHB}^+ \text{ attack}}(k, \epsilon, u, f)$ is non-negligible, then the UNLD problem can be efficiently solved.*

Proof. Refer Appendix B.

6 Conclusion And Future Work

In this paper, we have proven the hardness of a non-linear decoding problem that we call the UNLD problem and proposed the NLHB and NLHB⁺ authentication protocols, which are variants of the HB and HB⁺. These new protocols have better passive attack security than the HB and HB⁺ protocols. They have a low-complexity and are most suited for RFID tags and other low-cost devices deployed in scenarios with attack monitoring.

In the future, it would be interesting to see if the MIM attacks [9, 10] (part of a prevention-based attack model) on the HB family of protocols can be prevented by making appropriate changes to the NLHB protocol. This will give rise to a protocol that can be used in systems where the presence of attacks are not

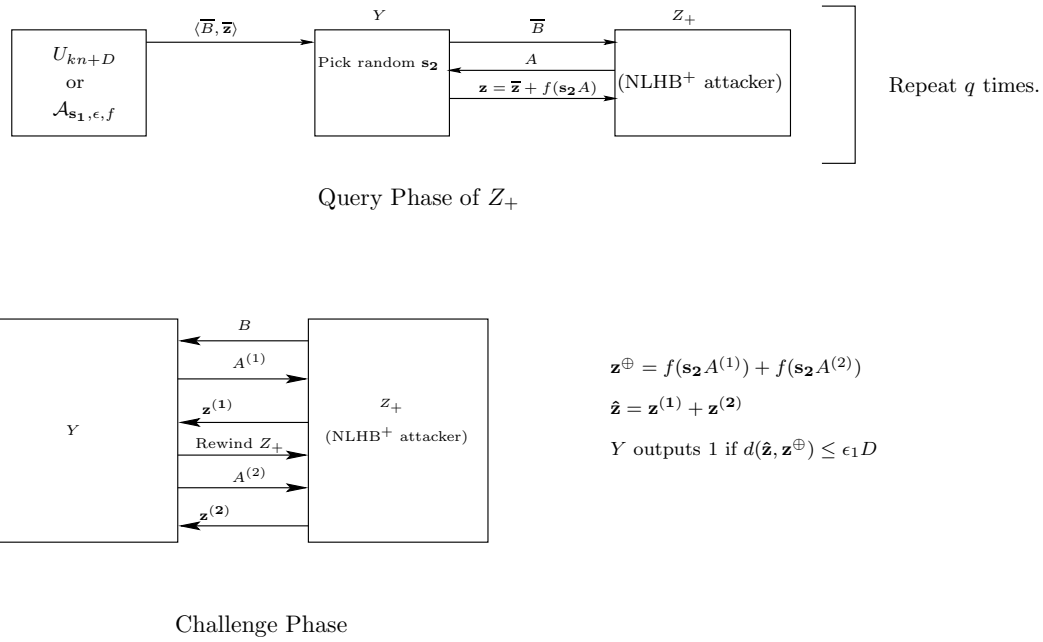


Fig. 7. Passing of Strings in Theorem 4 Proof

monitored. Another useful line of exploration would be to study if the NLHB protocol offers any advantage compared to other protocols in real channels, since noise is an intrinsic part of the protocol's design.

References

1. N.Hopper and M.Blum, "A Secure Human-Computer Authentication Scheme." Carnegie Mellon University, Tech. Rep. CMU-CS-00-139, 2000.
2. É. Leveil and P.-A. Fouque, "An Improved LPN Algorithm," in *Proceedings of SCN*, ser. LNCS, vol. 4116. Springer, 2006, pp. 348–359.
3. J. Carrijo, R. Tonicelli, H. Imai, and A. C. A. Nascimento, "A Novel Probabilistic Passive Attack on the Protocols HB and HB⁺," *IEICE Transactions*, pp. 658–662, 2009.
4. E. Berlekamp, R. McEliece, and H. van Tilborg, "On the Inherent Intractability of Certain Coding Problems," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, May 1978.
5. A. Juels and S. A. Weis, "Authenticating Pervasive Devices with Human Protocols," in *Proceedings of CRYPTO 2005*, ser. LNCS, vol. 3621. Springer, 2005, pp. 293–308.
6. A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model," *Journal of the ACM*, vol. 50, no. 4, pp. 506–519, 2003.
7. J. Katz and J. S. Shin, "Parallel and Concurrent Security of the HB and HB⁺ Protocols," in *Proceedings of EUROCRYPT 2006*, ser. LNCS, vol. 4004. Springer, 2006, pp. 73–87.
8. J. Katz and A. Smith, "Analyzing the HB and HB⁺ Protocols in the "Large Error" Case," Available from <http://eprint.iacr.org/2006/326.pdf>.
9. H. Gilbert, M. Robshaw, and H. Sibert, "Active attack against HB⁺: A Provably-Secure Lightweight Authentication Protocol," *IEE Electronics Letters*, vol. 41, no. 21, pp. 1169–1170, Oct. 2005.
10. K. Ouafi, R. Overbeck, and S. Vaudenay, "On the Security of HB[#] against a Man-in-the-Middle Attack," in *Proceedings of ASIACRYPT 2008*, ser. LNCS, vol. 5350. Springer, 2008, pp. 108–124.

A Formal Security Proof For NLHB In Passive Model

Theorem 2 : Reduction From UNLD Problem To Distinguishing $\mathcal{A}_{\mathbf{s},\epsilon,f}$ and U_{kn+D}

Suppose there exists a probabilistic polynomial-time algorithm Y taking q bitstrings of an unknown distribution (either $\mathcal{A}_{\mathbf{s},\epsilon,f}$ or U_{kn+D}) and outputting 0/1, running in time t , such that the probability of outputting 1 when its input is drawn from U_{kn+D} and when its input is drawn from $\mathcal{A}_{\mathbf{s},\epsilon,f}$ differ by at least δ , i.e

$$|Pr[\mathbf{s} \leftarrow \{0,1\}^k : Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = 1] - Pr[Y^{U_{kn+D}} = 1]| \geq \delta \quad (5)$$

Then there exists X taking $q' = O(q \cdot \delta^{-2} \log(k))$ bitstrings of $\mathcal{A}_{\mathbf{s},\epsilon,f}$ running in time $t' = O(t \cdot k \cdot \delta^{-2} \log(k))$ such that

$$Pr[\mathbf{s} \leftarrow \{0,1\}^k : X^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = \mathbf{s}] \geq \delta/4$$

Algorithm X does the following:

1. Pick $N = O(\delta^{-2} \log(k))$.
2. X chooses w coins for Y and uses these for the rest of the execution. ⁶
3. X runs $Y^{U_{kn+D}}(w)$ N times to obtain a bit string $\mathbf{b} = [b_1, \dots, b_N]$. Let $p = \frac{\text{wt}(\mathbf{b})}{N}$ be an estimate for the probability that $Y^{U_{kn+D}}$ outputs 1.
4. X obtains qN samples $\langle A_{v,j}, \mathbf{z}_{v,j} \rangle, j = 1, \dots, q; v = 1, \dots, N$ of distribution $\mathcal{A}_{\mathbf{s},\epsilon,f}$. (q samples per response bit from Y multiplied by the required N responses required from Y). For $i \in [k]$:
 - (a) Pick a random n -bit vector $\mathbf{c}_{v,j}$ for $j = 1, \dots, q; v = 1, \dots, N$. Modify $(A_{v,j})_i$, the i^{th} row of $A_{v,j}$, as $(A_{v,j})_i = (A_{v,j})_i + \mathbf{c}_{v,j}$ to get a modified matrix $A'_{v,j}$. Pass the modified instance $\langle A'_{v,j}, \mathbf{z}_{v,j} \rangle$ to Y for $v = 1, j = 1, \dots, q$ to obtain its response b'_1 . Repeat this for $v = 1, \dots, N$ to get the bit-string $\mathbf{b}^i = [b'_1, \dots, b'_N]$. Let $p_i = \frac{\text{wt}(\mathbf{b}^i)}{N}$ be an estimate of the probability that Y returns a 1 when the i^{th} row of $A_{v,j}; v \in [N], j \in [q]$ are modified.
 - (b) If $|p_i - p| \geq \delta/4$ set $s'_i = 0$, else set $s'_i = 1$.
5. Output $\mathbf{s}' = (s'_1, \dots, s'_k)$.

Note 1: There are three sources of randomness here. One is the randomness in the unknown key \mathbf{s} . The second is the randomness present in the decisions made by the existential algorithm Y . This randomness is denoted by the w coins chosen for Y at the beginning of the above algorithm. Both the key \mathbf{s} and the w coins are picked and held constant over the whole run of the algorithm. The third source of randomness comes from the picking of bitstrings from the distribution itself.

Note 2: The difference in the probabilities of Y outputting 1 for the two distributions in (5), is an averaged quantity. It is averaged over the key \mathbf{s} and also on the randomness in the algorithm Y itself. However, one run of the above algorithm only uses one instance of \mathbf{s} and w . So, instead of using these averaged probabilities in our analysis, we should use the probabilities associated with the particular key and randomness w used with this run of the algorithm Y . We will refer to the probability of Y outputting 1 when it uses these particular w coins as $Pr[Y^{U_{kn+D}}(w) = 1]$ (Similarly $Pr[Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}}(w) = 1]$), i.e we use the argument w to denote a particular set of decisions followed by Y .

Analysis of the Algorithm:

From the algorithm, we see that p is an estimate for $Pr[Y^{U_{kn+D}}(w) = 1]$. Further, if hyb_i denotes the distribution of the $(kn+D)$ bits passed to Y by X in step 4(a), then p_i is an estimate of $Pr[Y^{hyb_i}(w) = 1]$. We now prove that for the chosen value of $N = O(\delta^{-2} \log(k))$, p and p_i are very close estimates of these values.

⁶ The coins act as the source of randomness in Y . In other words, the choosing of this coins can be thought of as Y following one set of probabilistic decisions in its functioning out of the many possibilities.

Consider $\Pr[|\Pr[Y^{U_{kn+D}}(w) = 1] - p| \leq \delta/16]$, i.e the probability of the event the actual value of $\Pr[Y^{U_{kn+D}}(w) = 1]$ and its estimate are within $\delta/16$ of each other. For ease of readability, let us denote $\Pr[Y^{U_{kn+D}}(w) = 1]$ by Pr_U .

Accuracy of Estimates p_i and p :

We know that $p = \frac{\text{wt}(\mathbf{b})}{N}$. Each bit of \mathbf{b} follows a Bernoulli distribution with mean Pr_U . So, $\text{wt}(\mathbf{b})$ follows a Binomial distribution with mean NPr_U . So,

$$\begin{aligned} \Pr[|p - Pr_U| \leq \delta/16] &= \Pr[|\text{wt}(\mathbf{b}) - NPr_U| \leq N\delta/16], \\ &= \Pr[NPr_U - N\delta/16 \leq \text{wt}(\mathbf{b}) \leq NPr_U + N\delta/16], \\ &= 1 - \Pr[\text{wt}(\mathbf{b}) > NPr_U + N\delta/16] - \Pr[\text{wt}(\mathbf{b}) < NPr_U - N\delta/16]. \end{aligned}$$

By applying Chernoff bounds on this Binomial random variable, we have

$$\Pr[|p - Pr_U| \leq \delta/16] \geq 1 - \exp\left[-\frac{N\delta^2}{768Pr_U}\right] - \exp\left[-\frac{N\delta^2}{512Pr_U}\right]. \quad (6)$$

Now, we use $N = O(\delta^{-2}\log(k))$. Let d_1 be a large constant such that $N \leq d_1\delta^{-2}\log(k)$. Applying in (6), we get

$$\Pr[|p - Pr_U| \leq \delta/16] \geq 1 - \left(\frac{1}{k}\right)^{\frac{d_1}{768Pr_U}} - \left(\frac{1}{k}\right)^{\frac{d_1}{512Pr_U}}. \quad (7)$$

By similar reasoning, we also have,

$$\Pr[|Pr_{hi} - p_i| \leq \delta/16] \geq 1 - \left(\frac{1}{k}\right)^{\frac{d_1}{768Pr_{hi}}} - \left(\frac{1}{k}\right)^{\frac{d_1}{512Pr_{hi}}}, \quad (8)$$

where Pr_{hi} is used to denote $Pr[Y^{hybi}(w) = 1]$ for ease of readability. We know that, for two independent events $E1$ and $E2$, if $\Pr[E1] \geq 1 - a$ and $\Pr[E2] \geq 1 - b$, then $\Pr[E1 \cap E2] \geq 1 - a - b$. Applying this here, we see that $|p - Pr_U| \leq \delta/16$ and $|Pr_{hi} - p_i| \leq \delta/16$ (the latter for all i) hold simultaneously with probability

$$\geq 1 - \left\{ \left[\frac{1}{k}\right]^{\frac{d_1}{768Pr_U}} + \left[\frac{1}{k}\right]^{\frac{d_1}{512Pr_U}} \right\} - \sum_{i=1}^k \left\{ \left[\frac{1}{k}\right]^{\frac{d_1}{768Pr_{hi}}} + \left[\frac{1}{k}\right]^{\frac{d_1}{512Pr_{hi}}} \right\}.$$

Let $l = \min\left\{\frac{1}{768Pr_u}, \frac{1}{512Pr_u}, \frac{1}{768Pr_{h1}}, \dots, \frac{1}{768Pr_{hk}}, \frac{1}{512Pr_{h1}}, \dots, \frac{1}{512Pr_{hk}}\right\}$. Then the above expression can be lower-bounded as

$$\geq 1 - (2k + 2) \left(\frac{1}{k}\right)^{\frac{d_1}{l}} \geq 1 - 4k^{1-\frac{d_1}{l}}$$

By choosing d_1 sufficiently large, ($d_1 = 4l$, say), we have that (7) and (8) hold simultaneously with probability $\geq (1 - \frac{4}{k^3}) \geq \frac{1}{2}$ (for $k > 1$).

In summary we have that the following equations hold simultaneously with probability at least $\frac{1}{2}$.

$$|p - \Pr[Y^{U_{kn+D}}(w) = 1]| \leq \delta/16 \quad (9)$$

$$|p_i - \Pr[Y^{hybi}(w) = 1]| \leq \delta/16, 1 \leq i \leq k. \quad (10)$$

Suppose $s_i = 1$:

Now, consider the case where $s_i = 1$. By Lemma 1, in this case, $hyb_i = U_{kn+D}$. So, if both (9) and (10) hold, then for the case of $s_i = 1$, we have

$$|p_i - p| \leq 2\delta/16 = \delta/8. \quad (11)$$

Suppose $s_i = 0$:

Now consider $s_i = 0$. Then, since the i^{th} row of $A'_{v,j}$ never plays a role in the output, $hyb_i = \mathcal{A}_{\mathbf{s},\epsilon,f}$. Now let us bound $|p_i - p|$ in this case.

From the definition of Y , we have

$$|Pr[\mathbf{s} \leftarrow \{0,1\}^k : Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}} = 1] - Pr[Y^{U_{kn+D}} = 1]| \geq \delta.$$

This is a bound on the difference between the probabilities on an average. Using a standard averaging argument, we now derive a bound for the difference between the probabilities of Y outputting 1 in each case for the given instance of \mathbf{s} and w .

Lemma 2. *By a standard averaging argument, with probability $\geq \delta/2$ over the choice of \mathbf{s} and the random coins w , the following equation holds,*

$$|Pr[Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}}(w) = 1] - Pr[Y^{U_{kn+D}}(w) = 1]| \geq \delta/2, \quad (12)$$

where the probabilities inside the equation are over the randomness involved in picking bitstrings from the distributions.

Proof. We prove this Lemma at the end of this Theorem.

Since we know that when $s_i = 0$, $hyb_i = \mathcal{A}_{\mathbf{s},\epsilon,f}$, it follows that $Pr[Y^{hyb_i}(w) = 1] = Pr[Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}}(w) = 1]$. So, from (12), we have,

$$|Pr[Y^{hyb_i}(w) = 1] - Pr[Y^{U_{kn+D}}(w) = 1]| \geq \delta/2. \quad (13)$$

Rewriting (13), we have

$$\begin{aligned} \delta/2 &\leq |Pr[Y^{hyb_i}(w) = 1] - p_i + p_i - Pr[Y^{U_{kn+D}}(w) = 1] - p + p|, \\ &\leq |p_i - p| + |Pr[Y^{hyb_i}(w) = 1] - p_i| + |Pr[Y^{U_{kn+D}}(w) = 1] - p|, \\ &\leq |p_i - p| + \delta/16 + \delta/16, \end{aligned}$$

assuming (9) and (10) hold. Finally, this implies

$$|p_i - p| \geq \delta/2 - 2\delta/16 = 3\delta/8. \quad (14)$$

So, if $s_i = 0$, then $|p_i - p| \geq 3\delta/8$.

So, in Step 5 of the algorithm X , if $|p_i - p| \leq \frac{\delta}{4} = \frac{2\delta}{8}$, the estimated message bit is 1, else it is 0. Since (12) holds with probability at least $\frac{\delta}{2}$ and (9) and (10) hold with a further probability of .5, the probability that (11) and (14) hold is at least $\frac{\delta}{4}$. Hence, algorithm X succeeds with probability at least $\delta/4$. \square

Proof For Lemma 2 : Let R be the set of all possibilities for the key \mathbf{s} and the w coins. Then, by our definition of Y , we have,

$$\sum_{\mathbf{s}, w \in R} P[\mathbf{s}, w] |Pr[Y^{\mathcal{A}_{\mathbf{s},\epsilon,f}}(w) = 1] - Pr[Y^{U_{kn+D}}(w) = 1]| \geq \delta. \quad (15)$$

Let the subset $R' \subset R$ be the set such that $\forall r' = (\mathbf{s}', w') \in R'$, we have

$$|Pr[Y^{A_{\mathbf{s}', \epsilon, f}}(w') = 1] - Pr[Y^{U_{kn+D}}(w') = 1]| < \delta/2. \quad (16)$$

Then, in contradiction to Lemma 2, assume that the probability that r' is picked at random is atleast $1 - \delta/2$. Assuming that all r' are equally likely to be picked, this implies that $\sum_{(\mathbf{s}', w') \in R'} Pr(\mathbf{s}', w') \geq 1 - \delta/2$ and consequently, $\sum_{\mathbf{s}, w \in R \setminus R'} Pr(\mathbf{s}, w) < \delta/2$.

Now, splitting the left-hand-side (LHS) of (15) into summations over R' and $R' \setminus R$, we have

$$\begin{aligned} \text{LHS} &= \sum_{\mathbf{s}', w' \in R'} Pr[\mathbf{s}', w'] |Pr[Y^{A_{\mathbf{s}', \epsilon, f}}(w') = 1] - Pr[Y^{U_{kn+D}}(w') = 1]| \\ &+ \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w] |Pr[Y^{A_{\mathbf{s}, \epsilon, f}}(w) = 1] - Pr[Y^{U_{kn+D}}(w) = 1]|. \end{aligned}$$

By (16) and the fact that $|Pr[Y^{A_{\mathbf{s}, \epsilon, f}}(w) = 1] - Pr[Y^{U_{kn+D}}(w) = 1]| < 1$ (because it contains probability terms), we have

$$\begin{aligned} \text{LHS} &< (\delta/2) \sum_{\mathbf{s}', w' \in R'} Pr[\mathbf{s}', w'] + \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w](1), \\ &= (\delta/2)(1 - \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w]) + \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w]. \end{aligned}$$

Since we had $\delta \leq$ our original LHS from (15), this implies

$$\begin{aligned} \delta &< (\delta/2) + (1 - \delta/2) \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w], \\ &\Rightarrow \sum_{\mathbf{s}, w \in R \setminus R'} Pr[\mathbf{s}, w] > \delta/2, \end{aligned}$$

which contradicts our initial assumption about the set R' . So, by contradiction, Lemma 2 is true. \square

Theorem 3: Reduction From Distinguishing $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ and U_{kn+D} To Forging NLHB Protocol in Passive Model.

If $Adv_Z^{NLHB-attack}(k, \epsilon, u, f) = \delta$ is non-negligible for some polynomial time adversary Z , then the UNLD problem can be efficiently solved.

Algorithm for Theorem 3: Given access to Z which takes q bitstrings of $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ and runs in time t and forges the NLHB protocol with a passive attack, we construct an algorithm Y that takes $q + 1$ bitstrings from $\mathcal{A}_{\mathbf{s}, \epsilon, f}$, and can distinguish between strings drawn from U_{kn+D} and $\mathcal{A}_{\mathbf{s}, \epsilon, f}$. Y works like this.

1. Y has access to bitstrings from either $\mathcal{A}_{\mathbf{s}, \epsilon, f}$ or U_{kn+D} .
2. Y draws q strings $\langle A_i, \mathbf{z}_i \rangle_{i=0}^q$ from this distribution. This is passed on to Z .
3. Now Y obtains another sample pair $\langle \hat{A}, \hat{\mathbf{z}} \rangle$ from the distribution (the first kn bits of the bitstring drawn will represent \hat{A} in case of either input distribution) and challenges Z with \hat{A} . Let the received response be \mathbf{z}' .
4. Y outputs 1 if $\hat{\mathbf{z}}$ and \mathbf{z}' differ by atmost $u' = \epsilon''D$, i.e if $d(\mathbf{z}', \hat{\mathbf{z}}) \leq u'$, where ϵ'' is some constant such that $\epsilon' - 2\epsilon\epsilon' + \epsilon < \epsilon'' < \frac{1}{2}$.

Analysis of the algorithm: If Y 's input distribution is U_{kn+D} , the probability that Y outputs 1 is $p_U(1) = \sum_{i=0}^{u'} \binom{D}{i} 2^{-D}$. Since $\epsilon'' < .5$, $p_U(1)$ is negligible if D is large enough.

Let $\mathbf{z}^* = f(\mathbf{s}\hat{A})$. Let \mathbf{w} and \mathbf{e} be error vectors corresponding to \mathbf{z}' and $\hat{\mathbf{z}}$, i.e $\mathbf{z}' = \mathbf{z}^* + \mathbf{w}$ and $\hat{\mathbf{z}} = \mathbf{z}^* + \mathbf{e}$. Then, $d(\mathbf{z}', \mathbf{z}^*) \leq u$ implies that $\text{wt}(\mathbf{w}) \leq u$ and $d(\mathbf{z}', \hat{\mathbf{z}}) \leq u'$ implies that $\text{wt}(\mathbf{w} + \mathbf{e}) \leq u'$.

Consider the conditional probability $\Pr[d(\mathbf{z}', \hat{\mathbf{z}}) \leq u' \mid d(\mathbf{z}', \mathbf{z}^*) \leq u] = \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u]$. It is possible to show that

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] \geq \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = u] \quad (17)$$

We give a proof for (17) at the end of this proof. We will now consider the right-hand-side of (17) and prove that it is negligibly close to 1. The conditional expectation of $\text{wt}(\mathbf{w} + \mathbf{e})$ given $\text{wt}(\mathbf{w}) = u$ is given by

$$\begin{aligned} E[\text{wt}(\mathbf{w} + \mathbf{e}) \mid \text{wt}(\mathbf{w}) = u] &= u \cdot (1 - \epsilon) + (D - u)\epsilon \\ &= (\epsilon' - 2\epsilon\epsilon' + \epsilon)D \end{aligned}$$

Since $\epsilon'' > \epsilon' - 2\epsilon\epsilon' + \epsilon$, we see that the following Chernoff bound holds:

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) > (1 + \Delta)\mu \mid \text{wt}(\mathbf{w}) = u] \leq \left(\frac{\exp(\mu\Delta)}{(1 + \Delta)^{(1+\Delta)\mu}} \right),$$

where $\mu = (\epsilon' - 2\epsilon\epsilon' + \epsilon)D$ is the mean of the random variable $\text{wt}(\mathbf{w} + \mathbf{e})$ given that $\text{wt}(\mathbf{w}) = u$, $(1 + \Delta)\mu = \epsilon''D$, which imply that $\Delta = \frac{\epsilon''}{\epsilon' - 2\epsilon\epsilon' + \epsilon} - 1$.

So we have

$$\begin{aligned} &\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] \\ &\geq \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = u] \geq \left[1 - \left(\frac{\exp(\mu\Delta)}{(1 + \Delta)^{(1+\Delta)\mu}} \right) \right]. \end{aligned}$$

We also know that $\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u'] = \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] \Pr[\text{wt}(\mathbf{w}) \leq u]$. By the definition of the NLHB forger Z , we know that $\Pr[d(\mathbf{z}', \mathbf{z}^*) = \text{wt}(\mathbf{w}) \leq u] \geq (\delta + P_{FA})$, where P_{FA} denotes the probability of success of an attacker who responds with a random response (P_{FA} is known to be negligibly small at high D). So, we have

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u'] \geq (\delta + P_{FA}) \left[1 - \left(\frac{\exp(\mu\Delta)}{(1 + \Delta)^{(1+\Delta)\mu}} \right) \right]. \quad (18)$$

Consequently, the difference in the probabilities of Y outputting 1 for the two distributions is at least

$$(\delta + P_{FA}) \left[1 - \left(\frac{\exp(\mu\Delta)}{(1 + \Delta)^{(1+\Delta)\mu}} \right) \right] - \sum_{i=0}^{u'} \binom{D}{i} 2^{-D}. \quad (19)$$

Using suitable protocol parameters D, ϵ, ϵ' (say, $D = 1000, \epsilon = .25, \epsilon' = .348$ [2]), we see that the value in (19) is negligibly close to δ . This proves that Y can be constructed from Z . \square

Proof For (17) : We see that (by applying Bayes rule)

$$\begin{aligned} \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] &= \Pr[\text{wt}(\mathbf{w}) \leq u \mid \text{wt}(\mathbf{w} + \mathbf{e}) \leq u'] \frac{\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u']}{\Pr[\text{wt}(\mathbf{w}) \leq u]}, \\ &= \sum_{i=0}^u \Pr[\text{wt}(\mathbf{w}) = i \mid \text{wt}(\mathbf{w} + \mathbf{e}) \leq u'] \frac{\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u']}{\Pr[\text{wt}(\mathbf{w}) \leq u]}. \end{aligned}$$

Applying Bayes Rule again, the above expression reduces to

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] = \sum_{i=0}^u \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = i] \frac{\Pr[\text{wt}(\mathbf{w}) = i]}{\Pr[\text{wt}(\mathbf{w}) \leq u]}.$$

The random variable $\text{wt}(\mathbf{w} + \mathbf{e}) \mid \text{wt}(\mathbf{w}) = i$ is the sum of the bits of $\text{wt}(\mathbf{w} + \mathbf{e})$ and has a mean $\mu_i = (1 - \epsilon)i + (D - i)\epsilon$. Since the bits of $(\mathbf{w} + \mathbf{e})$ are independent, $(\text{wt}(\mathbf{w} + \mathbf{e}) \mid \text{wt}(\mathbf{w}) = i) \sim N(\mu_i, \sigma^2)$, where $\sigma^2 = D\epsilon(1 - \epsilon)$. So, the probability $\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = i]$ can be given by the Cumulative Distribution Function (CDF) $[1 - Q\left(\frac{u' - \mu_i}{\sigma}\right)]$ where the function $Q(\cdot)$ is the tail-probability of $N(0, 1)$ defined as $Q(\alpha) = \frac{1}{\sqrt{2\pi}} \int_{\alpha}^{\infty} e^{-\frac{x^2}{2}} dx$. Since Q -function is a decreasing function, and $\mu_i > \mu_{i-1}$, $\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = i]$ is a decreasing function of i . So, we have

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] \geq \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = u] \sum_{i=0}^u \frac{\Pr[\text{wt}(\mathbf{w}) = i]}{\Pr[\text{wt}(\mathbf{w}) \leq u]}.$$

This implies that

$$\Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) \leq u] \geq \Pr[\text{wt}(\mathbf{w} + \mathbf{e}) \leq u' \mid \text{wt}(\mathbf{w}) = u]. \quad (20)$$

□

From Theorems 1 to 3, we can see that, if UNLD is hard, then it is hard to forge a Prover of the NLHB protocol in polynomial-time, making NLHB computationally secure in the passive attack model.

B Security Proof For NLHB⁺ In DET Model:

Theorem 4: Reduction From UNLD Problem To Active Attack on NLHB⁺: *If for some polynomial-time adversary Z_+ , $\text{Adv}_{Z_+}^{\text{NLHB}^+ \text{ attack}}(k, \epsilon, u, f) = \delta$ is non-negligible, the UNLD problem can be efficiently solved.*

To prove this, we show how to build the algorithm Y that can differentiate between distributions U_{kn+D} and $A_{\mathbf{s}_1, \epsilon, f}$ (for secret \mathbf{s}_1) using access to a NLHB⁺ adversary Z_+ in the “DET” model. This proof strategy is based on [8].

Algorithm for Y :

1. Y chooses \mathbf{s}_2 at random from $\{0, 1\}^k$. During the query phase of Z_+ , Y draws the bitstring $\langle \bar{B}, \bar{\mathbf{z}} \rangle$ (as usual, irrespective of the input distribution, the first kn bits will form \bar{B}) from its unknown input distribution (U_{kn+D} or $A_{\mathbf{s}_1, \epsilon, f}$) and passes \bar{B} to Z_+ . Z_+ replies with challenge A . In response, Y sends $\mathbf{z} = \bar{\mathbf{z}} + f(\mathbf{s}_2 A)$ to Z_+ . This is repeated q times.
2. In its challenge phase, Z_+ sends a matrix B as blinding matrix to Y . Y challenges Z_+ with random matrix $A^{(1)}$ and receives response $\mathbf{z}^{(1)}$ from Z_+ . Now, Y rewinds Z_+ and challenges it with another random matrix $A^{(2)}$ and receives $\mathbf{z}^{(2)}$ in response.
3. Let $\mathbf{z}^{\oplus} = \mathbf{z}^{(1)} + \mathbf{z}^{(2)}$. Further, let $\hat{\mathbf{z}} = f(\mathbf{s}_2 A^{(1)}) + f(\mathbf{s}_2 A^{(2)})$. Y outputs 1 if \mathbf{z}^{\oplus} and $\hat{\mathbf{z}}$ differ in fewer than $u' = \epsilon_1 D$ entries. (ϵ_1 to be defined).

Analysis of the Algorithm: When Y 's input is U_{kn+D} , $\bar{\mathbf{z}}$ is uniformly distributed. Hence $\mathbf{z} = \bar{\mathbf{z}} + f(\mathbf{s}_2 A)$ is also uniformly distributed and independent of \mathbf{s}_2 . So no information about \mathbf{s}_2 reaches Z_+ in the query phase. This means that, as far as Z_+ is concerned, $\hat{\mathbf{z}}$ is uniformly distributed in the random code $C = \{f(\mathbf{s}_2 A^{(1)}) + f(\mathbf{s}_2 A^{(2)})\}_{\mathbf{s}_2}$. Now, we show that $\Pr[d(\mathbf{z}^{\oplus}, \hat{\mathbf{z}}) \leq \epsilon_1 D]$ is negligibly small for large D .

Consider the Hamming Ball B of radius $\epsilon_1 D$ centred at \mathbf{z}^\oplus . Let X be the number of codewords of C present in this Hamming Ball. When the matrices $A^{(1)}$ and $A^{(2)}$ are picked, they are picked uniformly at random. This means, because of Property 3 of f (uniform inputs \Rightarrow uniform outputs), the vectors in code C form a random code. We now apply the Markov Inequality on X .

$$\Pr[X \geq p] \leq \frac{E(X)}{p}, \quad (21)$$

where $E(X)$ is the mean of X . Now consider $\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D]$. We see that

$$\begin{aligned} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] &= \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p] \Pr[X < p] + \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X \geq p] \Pr[X \geq p] \\ &\geq \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p] \Pr[X < p] \end{aligned} \quad (22)$$

Consider $\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p]$. This can be written as

$$\begin{aligned} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p] &= \Pr[X < p \mid d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \frac{\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D]}{\Pr[X < p]}, \\ &= \sum_{i=0}^{p-1} \Pr[X = i \mid d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \frac{\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D]}{\Pr[X < p]}, \\ &= \sum_{i=0}^{p-1} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X = i] \frac{\Pr[X = i]}{\Pr[X < p]}. \end{aligned}$$

Notice that the quantity $\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X = i]$ will decrease with increase in i . This is because, with more codewords of C within the Hamming ball B , the higher is the chance that $\hat{\mathbf{z}}$ lies within the Hamming Ball B , and so, higher is the chance that the distance between $\hat{\mathbf{z}}$ and \mathbf{z}^\oplus is within $\epsilon_1 D$. So, we can write

$$\begin{aligned} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p] &\geq \sum_{i=0}^{p-1} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X = p] \frac{\Pr[X = i]}{\Pr[X < p]}, \\ &= \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X = p]. \end{aligned}$$

So, we have from (22) that

$$\begin{aligned} \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] &\geq \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X < p] \Pr[X < p], \\ &\geq \Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D \mid X = p] \Pr[X < p], \\ &= \Pr[\hat{\mathbf{z}} \notin B \mid X = p] \Pr[X < p]. \end{aligned} \quad (23)$$

We know from the Markov inequality in (21), that $\Pr[X < p]$ is lower bounded by $\left(1 - \frac{E(X)}{p}\right)$. The mean number of codewords from C , which are part of the Hamming Ball B is given by $E(X) = \left(\frac{|B|}{2^k}\right) 2^k$. So (23) becomes

$$\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \geq \left(1 - \frac{|B| 2^{k-D}}{p}\right). \quad (24)$$

Out of the 2^k codewords of C , the probability that $\hat{\mathbf{z}}$ is one of the p codewords in B is given by $\frac{p}{2^k}$. So, the probability that $\hat{\mathbf{z}}$ does not belong to the Hamming ball B when it is known that B has exactly p codewords of C , is given by $\left(1 - \frac{p}{2^k}\right)$. So,

$$\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \geq \left(1 - \frac{p}{2^k}\right) \left(1 - \frac{|B| 2^{k-D}}{p}\right).$$

Pick $p = 2^{3k/4}$, say. Then

$$\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \geq \left(1 - 2^{-k/4}\right) \left(1 - |B| 2^{\frac{k}{4}-D}\right).$$

We notice that $|B|$, the number of vectors in a Hamming Ball of radius $\epsilon_1 D$ is given by $|B| = \sum_{i=0}^{\epsilon_1 D} \binom{D}{i}$. So,

$$\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) > \epsilon_1 D] \geq \left(1 - 2^{-k/4}\right) \left(1 - 2^{\frac{k}{4}-D} \sum_{i=0}^{\epsilon_1 D} \binom{D}{i}\right).$$

Since $\epsilon_1 < \frac{1}{2}$, this bound tends to 1 asymptotically with D . So, the probability $\Pr[d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) \leq \epsilon_1 D]$ becomes negligibly small. So, in case the input distribution to Y is U_{kn+D} , the probability of Y outputting 1 is negligible.

When Y 's input distribution is $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$ for randomly chosen \mathbf{s}_1 , Y perfectly simulates the NLHB⁺ protocol to Z_+ during the query phase. Let w denote the randomness involved in simulating the query phase of Z_+ , which includes Z_+ 's randomness, the randomness in choosing $(\mathbf{s}_1, \mathbf{s}_2)$, and the randomness in responding to Z_+ 's queries. Let $(\delta_w + P_{FA})$ be the probability that Z_+ successfully impersonates the Prover in second phase when the randomness is w . Then Z_+ correctly replies to both queries $A^{(1)}$ and $A^{(2)}$ with probability $(\delta_w + P_{FA})^2$. The overall probability that Z_+ successfully responds to both sets of queries is

$$\mathbb{E}_w((\delta_w + P_{FA})^2) \geq (\mathbb{E}_w(\delta_w + P_{FA}))^2 = (\delta + P_{FA})^2 \quad (25)$$

using Jensen's inequality and \mathbb{E}_w denotes expectation over w . Conditioned on this event, we show that for an appropriate ϵ_1 , \mathbf{z}^\oplus and $\hat{\mathbf{z}}$ differ by fewer than u' entries with a constant probability (proven below). So Y outputs 1 with probability $\Omega((\delta + P_{FA})^2)$, which implies that Y can distinguish U_{kn+D} and $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$ with non-negligible probability. This concludes the proof of Theorem 4. \square

Pf. for \mathbf{z}^\oplus and $\hat{\mathbf{z}}$ differing by $\leq u'$ entries: Set $\frac{1}{2} > \epsilon_1 > \frac{1}{2}(1 - (1 - 2\epsilon')^2)$. Fixing all randomness, let f_{Z_+} denote the mapping that the adversary does from a challenge matrix A to the response \mathbf{z} in the second phase. Since we are looking at the process after B has been fixed, B is not an argument to the function f_{Z_+} . Let $f_{correct}$ denote $f(\mathbf{s}_1 A) + f(\mathbf{s}_2 B)$. Define $\Delta(A) = f_{Z_+}(A) + f_{correct}(A)$. We say that A is a *good* query matrix if $\text{wt}(\Delta(A)) \leq u$, i.e if Z_+ successfully impersonates the Prover for that matrix. Let D_Δ denote the distribution of $\Delta(A)$ over all good query matrices. Note that by definition, for all $\Delta(A)$ in D_Δ , $\text{wt}(\Delta(A)) \leq u$.

Let $\Delta^{(1)} = \Delta(A^{(1)})$ and $\Delta^{(2)} = \Delta(A^{(2)})$. Then,

$$\Delta^{(1)} + \Delta^{(2)} = f_{Z_+}(A^{(1)}) + f_{Z_+}(A^{(2)}) + f_{correct}(A^{(1)}) + f_{correct}(A^{(2)}).$$

Using $f_{correct}(A^{(1)}) + f_{correct}(A^{(2)}) = f(\mathbf{s}_2 A^{(1)}) + f(\mathbf{s}_2 A^{(2)}) = \hat{\mathbf{z}}$ and $f_{Z_+}(A^{(1)}) + f_{Z_+}(A^{(2)}) = \mathbf{z}^\oplus$, we see that $d(\Delta^{(1)}, \Delta^{(2)}) \leq u'$ whenever $d(\mathbf{z}^\oplus, \hat{\mathbf{z}}) \leq u'$. We now analyse the probability that $d(\Delta^{(1)}, \Delta^{(2)}) \leq u'$.

Using arguments based on the Johnson bound as in [8], we can show that $\Pr[d(\Delta^{(1)}, \Delta^{(2)}) \leq u'] > \frac{1}{2c^2}$, where $c = \frac{1 - \delta_{eps}}{\gamma^2 - \delta_{eps}} + 1$, $\delta_{eps} = 1 - 2\epsilon_1$ and $\gamma = 1 - 2\epsilon'$. So Y outputs 1 with probability at least $\frac{1}{2c^2}(\delta + P_{FA})^2$ when the input distribution is $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$. \square

So, the difference in probabilities of Y in the proff of Theorem 4 outputting a 1 for the two distributions $\mathcal{A}_{\mathbf{s}_1, \epsilon, f}$ and U_{kn+D} is at least

$$\left(\frac{1}{2c^2}(\delta + P_{FA})^2\right) - 2^{-k/4} - 2^{\frac{k}{4}-D} \sum_{i=0}^{\epsilon_1 D} \binom{D}{i} + 2^{-D} \sum_{i=0}^{\epsilon_1 D} \binom{D}{i}. \quad (26)$$

We see that this difference in probabilities tends to the non-negligible quantity $\frac{1}{2c^2}\delta^2$ asymptotically with D (and for fixed reasonably large k).

Thus Theorems 2 and 4 together show a reduction from the UNLD problem to the problem of active attack on NLHB⁺ protocol. Since the UNLD problem is known to be hard now, the active attack problem is also hard.

C Proof For Uniformity of Function f

Theorem : f is a Balanced Function: If the input to the function f is uniformly distributed, so is its output.

Proof We first prove that each bit of the output is balanced. For this, we consider $\Pr[y_i = 1]$.

$$\begin{aligned}\Pr[y_i = 1] &= \Pr[x_i + g(x_{i+1}, \dots, x_{i+p}) = 1], \\ &= \frac{1}{2}\Pr[g(x_{i+1}, \dots, x_{i+p}) = 1 \mid x_i = 0] + \frac{1}{2}\Pr[g(x_{i+1}, \dots, x_{i+p}) = 0 \mid x_i = 1].\end{aligned}$$

Since the input vector is uniform, the bits of \mathbf{x} are independent. So, this is equal to

$$\begin{aligned}&= \frac{1}{2}\Pr[g(x_{i+1}, \dots, x_{i+p}) = 1] + \frac{1}{2}\Pr[g(x_{i+1}, \dots, x_{i+p}) = 0], \\ &= \frac{1}{2}.\end{aligned}\tag{27}$$

So each bit of the output is balanced. Now, we use this to prove our theorem. To this end, we first define the following vectors. Let $\mathbf{y}^i = [y_{D-i+1}, \dots, y_D]$ be the vector containing the last i bits of \mathbf{y} . So $\mathbf{y}^D = \mathbf{y}$. Let $\mathbf{a} = [a_1, \dots, a_D]$ be an arbitrary constant D -bit vector. We also define $\mathbf{a}^i = [a_{D-i+1}, \dots, a_D]$ similar to \mathbf{y}^i . Now consider the probability $\Pr[\mathbf{y}^i = \mathbf{a}^i]$.

$$\begin{aligned}\Pr[\mathbf{y}^i = \mathbf{a}^i] &= \Pr[\mathbf{y}^i = \mathbf{a}^i \mid x_{D-i+1} = 0]\Pr[x_{D-i+1} = 0] \\ &\quad + \Pr[\mathbf{y}^i = \mathbf{a}^i \mid x_{D-i+1} = 1]\Pr[x_{D-i+1} = 1].\end{aligned}$$

Since the input is uniformly distributed, this is equal to

$$\begin{aligned}&= \frac{1}{2}\Pr[\mathbf{y}^i = \mathbf{a}^i \mid x_{D-i+1} = 0] + \frac{1}{2}\Pr[\mathbf{y}^i = \mathbf{a}^i \mid x_{D-i+1} = 1], \\ &= \frac{1}{2}\Pr[g(x_{D-i+2}, \dots, x_{D-i+p+1}) = a_{D-i+1}, \mathbf{y}^{i-1} = \mathbf{a}^{i-1} \mid x_{D-i+1} = 0], \\ &\quad + \frac{1}{2}\Pr[g(x_{D-i+2}, \dots, x_{D-i+p+1}) = a_{D-i+1} + 1, \mathbf{y}^{i-1} = \mathbf{a}^{i-1} \mid x_{D-i+1} = 1].\end{aligned}\tag{28}$$

We point out that in the vector \mathbf{y}^i , only the bit y_{D-i+1} is dependent on x_{D-i+1} . Since both $g(x_{D-i+2}, \dots, x_{D-i+p+1})$ and \mathbf{y}^{i-1} are independent of x_{D-i+1} , we can remove the conditioning from the above equation. So the above expression becomes,

$$\begin{aligned}&\frac{1}{2} (\Pr[g(x_{D-i+2}, \dots, x_{D-i+p+1}) = a_{D-i+1}, \mathbf{y}^{i-1} = \mathbf{a}^{i-1}] \\ &\quad + \Pr[g(x_{D-i+2}, \dots, x_{D-i+p+1}) = a_{D-i+1} + 1, \mathbf{y}^{i-1} = \mathbf{a}^{i-1}]).\end{aligned}\tag{29}$$

Now $g(x_{D-i+2}, \dots, x_{D-i+p+1})$ takes binary values 0 and 1. So, by summing the joint probability of $g(x_{D-i+2}, \dots, x_{D-i+p+1})$ and \mathbf{y}^{i-1} over these values, we are effectively finding the marginal probability of \mathbf{y}^{i-1} . So, from the expressions in Eqn. 2 and 3, we have

$$\Pr[\mathbf{y}^i = \mathbf{a}^i] = \frac{1}{2} (\Pr[\mathbf{y}^{i-1} = \mathbf{a}^{i-1}]). \quad (30)$$

Plugging $i = D$ in the above equation, and expanding, we have

$$\begin{aligned} \Pr[\mathbf{y}^D = \mathbf{a}^D] &= \frac{1}{2} (\Pr[\mathbf{y}^{D-1} = \mathbf{a}^{D-1}]) \\ &= \frac{1}{2^2} (\Pr[\mathbf{y}^{D-2} = \mathbf{a}^{D-2}]) \\ &\quad \vdots \\ &= \frac{1}{2^{D-1}} (\Pr[\mathbf{y}^1 = \mathbf{a}^1]) = \frac{1}{2^{D-1}} (\Pr[y_D = a_D]) \\ &= \frac{1}{2^D} \end{aligned} \quad (31)$$

from (27). Since this proof holds for any \mathbf{a}^i , the output of f is uniformly distributed. \square

D Notations

- All vectors are denoted in bold letters. Scalars are denoted in normal text.
- $]0, \frac{1}{2}[$ denotes open-interval from 0 to $\frac{1}{2}$.
- $\{0, 1\}^x$ denotes the space of all binary vectors of length x .
- $\{0, 1\}^{x \times y}$ denotes the space of all binary matrices of size $x \times y$.
- $\text{wt}(\mathbf{x})$ denotes the Hamming weight of the binary vector \mathbf{x} . This is equal to the number of non-zero entries in \mathbf{x} .
- $d(\mathbf{x}, \mathbf{y})$ denotes the Hamming distance between binary vectors \mathbf{x} and \mathbf{y} . This is equal to the number of places where \mathbf{x} and \mathbf{y} differ.
- $GF(2)$ denotes Galois Field with two entries.
- In this paper, $+$ is used to denote XOR addition which is the addition over $GF(2)$.
- $\binom{n}{k} = \frac{n!}{k!(n-k)!}$, where $n!$ denotes factorial.
- $U \leftarrow$ denotes "picked uniformly at random from".
- When a distribution is superscripted over an algorithm, (for e.g. X^A) this means that the algorithm X has input following the distribution A .
- $\mathcal{A}_{s, \epsilon, f}$ denotes the distribution followed by the $(kn + D)$ -length bitstrings that form the transcript of one authentication session between honest NLHB prover and verifier, for a shared secret \mathbf{s} .
- U_{kn+D} denotes the distribution of uniformly distributed $(kn + D)$ -length bitstrings.
- For a set R and its subset $R' \subset R$, $R \setminus R'$ denotes the set containing all the elements in R that are not in R' .