

 Open access • Journal Article • DOI:10.1007/S00145-016-9247-3

Efficient Authentication from Hard Learning Problems — [Source link](#)

[Eike Kiltz](#), [Krzysztof Pietrzak](#), [Daniele Venturi](#), [David Cash](#) ...+1 more authors

Institutions: [Ruhr University Bochum](#), [Institute of Science and Technology Austria](#), [University of Trento](#), [Rutgers University](#) ...+1 more institutions

Published on: 01 Oct 2017 - [Journal of Cryptology](#) (Springer US)

Topics: [Authentication protocol](#), [Data Authentication Algorithm](#), [Otway–Rees protocol](#) and [Message authentication code](#)

Related papers:

- [Secure Human Identification Protocols](#)
- [A Protocol for Provably Secure Authentication of a Tiny Entity to a High Performance Computing One](#)
- [Authenticating pervasive devices with human protocols](#)
- [HB # : increasing the security and efficiency of HB +](#)
- [On Authentication Schemes Using Polynomials Over Non Commutative Rings](#)

Share this paper:    

View more about this paper here: <https://typeset.io/papers/efficient-authentication-from-hard-learning-problems-5v9xf8xw8q>

Efficient Authentication from Hard Learning Problems[†]

Eike Kiltz*

Ruhr-Universität Bochum, Bochum, Germany

Krzysztof Pietrzak**

IST Austria, Klosterneuburg, Austria

Daniele Venturi

University of Trento, Trento, Italy
daniele.venturi@unitn.it

David Cash

Rutgers University, New Brunswick, NJ, USA

Abhishek Jain

Johns Hopkins University, Baltimore, MD, USA

Communicated by Jonathan Katz.

Received 25 August 2014 / Revised 12 September 2016

Online publication 9 December 2016

Abstract. We construct efficient authentication protocols and message authentication codes (MACs) whose security can be reduced to the learning parity with noise (LPN) problem. Despite a large body of work—starting with the HB protocol of Hopper and Blum in 2001—until now it was not even known how to construct an efficient authentication protocol from LPN which is secure against man-in-the-middle attacks. A MAC implies such a (two-round) protocol.

Keywords. Authentication protocols, Message authentication, Hard learning problems.

[†] A preliminary version of this paper is published in the proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques—Eurocrypt 2011 [33]. This is the full version.

* Eike Kiltz: Supported by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research.

** Krzysztof Pietrzak: Supported by the European Research Council, ERC Starting Grant (259668-PSPC) and ERC Consolidator Grant (682815-TOCNeT).

1. Introduction

Authentication is among the most basic and important cryptographic tasks. In the present paper we construct efficient (secret key) authentication schemes from the assumption that the *learning parity with noise* (LPN) problem is hard. Informally, this problem asks to distinguish noisy inner products from random. We construct the first efficient message authentication codes (MACs) from LPN, but also simpler and more efficient two-round authentication protocols that achieve a notion called active security. Prior to our work, the only known way to construct an LPN-based MAC was via a relatively inefficient generic transformation [23] (that works with any pseudorandom generator), and all interactive LPN-based protocols with security properties similar to our new protocol required at least three rounds and had a loose security reduction. Our constructions and techniques diverge significantly from prior work in the area and will hopefully be of independent interest.

The pursuit of LPN-based authentication is motivated by two disjoint concerns, one theoretical and one practical. On the theoretical side, the LPN problem provides an attractive basis for provable security [3,5,6,27,32,41]. It is closely related to the well-studied problem of decoding random linear codes, and unlike most number-theoretical problems used in cryptography, the LPN problem does not succumb to known quantum algorithms. On the practical side, LPN-based authentication schemes are strikingly efficient, requiring relatively few bit-level operations. Indeed, in their original proposal, Hopper and Blum [27] suggested that humans could perform the computation in their provably secure scheme, even with realistic parameters.

Each of our theoretical and practical motivations, on its own, would be sufficiently interesting for investigation, but together the combination is particularly compelling. LPN-based authentication is able to provide a theoretical improvement in terms of provable security *in addition to* providing better efficiency than approaches based on more classical symmetric techniques that are not related to hard problems. Usually we trade one benefit for the other, but here we hope to get the best of both worlds.

Before describing our contributions in more detail, we start by recalling authentication protocols, the LPN problem, and some of the prior work on which we build.

Authentication Protocols. An authentication protocol is a (shared-key) protocol where a prover \mathcal{P} authenticates itself to a verifier \mathcal{V} . We recall some of the common definitions for security against impersonation attacks. A *passive attack* proceeds in two phases, where in the first phase the adversary eavesdrops on several interactions between \mathcal{P} and \mathcal{V} , and then attempts to cause \mathcal{V} to accept in the second phase (where \mathcal{P} is no longer available). In an *active attack*, the adversary is additionally allowed to interact with \mathcal{P} in the first phase. The strongest and most realistic attack model is a *man-in-the-middle attack* (MIM), where the adversary can arbitrarily interact with \mathcal{P} and \mathcal{V} (with polynomially many concurrent executions allowed) and wins if it can make \mathcal{V} accept in any execution where at least one message was changed.

The LPN Problem. For $\tau < 1/2$ and a vector $\mathbf{x} \in \mathbb{Z}_2^\ell$, define the distribution $\Lambda_{\tau,\ell}(\mathbf{x})$ on $\mathbb{Z}_2^\ell \times \mathbb{Z}_2$ by $(\mathbf{r}, \mathbf{r}^\top \mathbf{x} + e)$, where $\mathbf{r} \in \mathbb{Z}_2^\ell$ is uniformly random and $e \in \mathbb{Z}_2$ is selected accord-

ing to Ber_τ , the Bernoulli distribution over \mathbb{Z}_2 with parameter τ (i.e., $\Pr[e = 1] = \tau$). The $\text{LPN}_{\tau,\ell}$ problem is to distinguish an oracle returning samples from $\Lambda_{\tau,\ell}(\mathbf{x})$, where $\mathbf{x} \in \mathbb{Z}_2^\ell$ is random and fixed, from an oracle returning uniform samples. It was shown by Blum et al. [5] that this is equivalent to the search version of LPN, where one needs to find \mathbf{x} given oracle access to $\Lambda_{\tau,\ell}(\mathbf{x})$ (cf. [31, Thm.2] for precise bounds). We note that the search variant of LPN is efficiently solvable when there is no noise (i.e., when $\tau = 0$) using Gaussian elimination, for this we just need ℓ samples with linearly independent \mathbf{r} 's. The best algorithms take time $2^{\ell/\log \ell}$ when $\tau > 0$ is treated as a constant [6–9, 24, 34, 46].

Authentication Protocols from LPN. Starting with the work of Hopper and Blum [27], several authentication protocols based on the LPN problem have been proposed. Their original elegant protocol is simple enough for us to recall right away. The shared secret key is a binary vector $\mathbf{s} \in \mathbb{Z}_2^\ell$. The interaction consists of two messages. First \mathcal{V} sends a random challenge $\mathbf{r} \in \mathbb{Z}_2^\ell$, and then \mathcal{P} answers with the bit $z = \mathbf{r}^\top \mathbf{s} + e$, where $e \in \mathbb{Z}_2$ is sampled according to Ber_τ . Finally, the verifier accepts if $z = \mathbf{r}^\top \mathbf{s}$.

This basic protocol has a large completeness error τ (as \mathcal{V} will reject if $e = 1$) and soundness error $1/2$ (as a random \mathbf{r} , z satisfies $\mathbf{r}^\top \mathbf{s} = z$ with probability $1/2$). This can be reduced via sequential or parallel composition. The parallel variant, denoted HB , is illustrated in Fig. 1 (we represent several \mathbf{r} with a matrix \mathbf{R} and the noise bits are now arranged in a vector \mathbf{e}). The verifier accepts if at least a τ' fraction (where $\tau < \tau' < 1/2$) of the n basic authentication steps is correct.

The two-round HB protocol is provably secure against passive attacks, but efficient active attacks are known against it. This is unsatisfying because in several scenarios an adversary *will* be able to mount an active attack. Subsequently, Juels and Weis [28] proposed an efficient three-round variant of HB , called HB^+ , and proved it secure against active attacks. Again the error can be reduced by sequential repetition, and as shown by Katz, Shin and Smith via a non-trivial analysis, parallel repetition works as well [30, 31]. The protocol (in its parallel repetition variant) is illustrated in Fig. 2.

Gilbert et al. [20] showed that HB^+ can be broken by a MIM attack. Several variants HB^{++} [11], HB^* [16], HB-MP [37] were proposed to prevent the particular attack from [20], but all of them were later shown to be insecure [21]. In [22], a variant $\text{HB}^\#$ was presented which provably resists the particular attack from [20], but was shown susceptible to a more general MIM attack [38]. However, no improvements in terms of round complexity, security or tightness of the reduction over HB^+ were achieved: Three-round protocols achieving active security $\sqrt{\epsilon}$ (assuming LPN is ϵ -hard) are the state of the art.

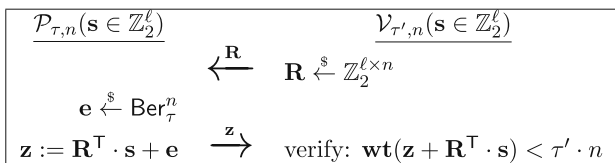


Fig. 1. The HB protocol, secure against passive attacks.

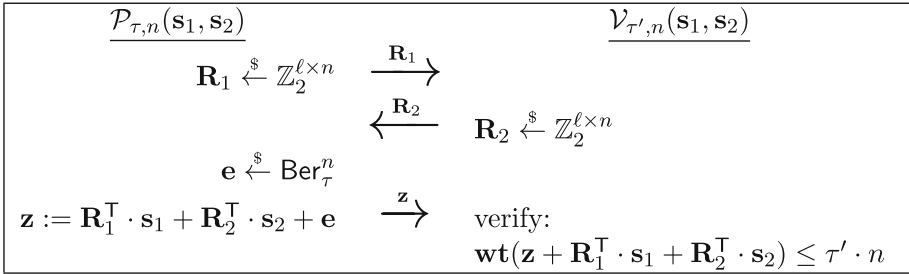


Fig. 2. The HB^+ protocol, secure against active attacks.

1.1. Our Contribution

We provide new constructions of authentication protocols and even MACs from LPN. Our first contribution is a *two*-round authentication protocol secure against active adversaries (this is mentioned as an open problem in [28]) which moreover has a tight security reduction (an open problem mentioned in [31]). As a second contribution, we build two efficient MACs and thus also get two-round authentication protocols secure against MIM attacks, from the LPN assumption. Unlike some previous proposals, our constructions are not ad hoc, and we give a reduction to the LPN problem. Our authentication protocol is roughly as efficient as the HB^+ protocol but has twice the key length. Our MACs perform roughly the same computation as the authentication protocol plus one evaluation of a pairwise independent permutation of an $\approx 2\ell$ -bit domain, where ℓ is the length of an LPN secret.

Two-Round Authentication with Active Security. Our first contribution is a two-round authentication protocol which we prove secure against *active* attacks assuming the hardness of the LPN problem. Our protocol diverges considerably from all previous HB -type protocols [22,27,28,31] and runs counter to the intuition that the only way to efficiently embed the LPN problem into a two-round protocol is via an HB -type construction.

We now sketch our protocol. In HB and its variants, the prover must compute LPN samples of the form $\mathbf{R}^T \cdot \mathbf{s} + \mathbf{e}$, where \mathbf{R} is the challenge chosen by the verifier in the first message. We take a different approach. Instead of sending \mathbf{R} , we now let the verifier choose a random subset of the bits of \mathbf{s} to act as the “session key” for this interaction. It represents this subset by sending a binary vector $\mathbf{v} \in \mathbb{Z}_2^\ell$ that acts as a “bit selector” of the secret \mathbf{s} , and we write $\mathbf{s}_{\downarrow \mathbf{v}}$ for the sub-vector of \mathbf{s} which is obtained by deleting all bits from \mathbf{s} where \mathbf{v} is 0. (E.g., if $\mathbf{s} = (1, 1, 1, 0, 0, 0)^T$, $\mathbf{v} = (0, 1, 1, 1, 0, 0)^T$ then $\mathbf{s}_{\downarrow \mathbf{v}} = (1, 1, 0)^T$.) The prover then picks \mathbf{R} by *itself* and computes noisy inner products of the form $\mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e}$. Curiously, allowing the verifier to choose which bits of \mathbf{s} to use in each session is sufficient to prevent active attacks. We only need to add a few sanity checks that no pathological \mathbf{v} or \mathbf{R} was sent by an active adversary.

Our proof relies on the recently introduced *subspace LPN problem* [40]. In contrast to the active attack security proof of HB^+ [31], our proof does not use any rewinding techniques. Avoiding rewinding has at least two advantages. First, the security reduction

becomes tight. Second, the proofs also work in a quantum setting: Our protocol is secure against quantum adversaries assuming LPN is secure against such adversaries. As first observed by van de Graaf [43], classical proofs using rewinding in general do not translate to the quantum setting (cf. [45] for a more recent discussion). Let us emphasize that this only means that there is no security proof for HB^+ in the quantum setting, but we do not know if a quantum attack actually exists.

MAC and Man-in-the-Middle Security. In Sect. 4, we give two constructions of message authentication codes (MACs) that are secure (formally, unforgeable under chosen message attacks) assuming that the LPN problem is hard. It is well known that a MAC implies a two-round MIM secure authentication protocol: The verifier chooses a random message as challenge, and the prover returns the MAC on the message.

As a first attempt, let us try to view our authentication protocol as a MAC. That is, a MAC tag is of the form $\phi = (\mathbf{R}, \mathbf{z} = \mathbf{R}^\top \cdot f_{\mathbf{s}}(m) + \mathbf{e})$, where the secret key derivation function $f_{\mathbf{s}}(m) \in \mathbb{Z}_2^\ell$ first uniquely encodes the message m into $\mathbf{v} \in \mathbb{Z}_2^{2\ell}$ of weight ℓ and then returns $\mathbf{s}_{\downarrow \mathbf{v}}$ by selecting ℓ bits from secret \mathbf{s} , according to \mathbf{v} . However, this MAC is not secure: Given a MAC tag $\phi = (\mathbf{R}, \mathbf{z})$ an adversary can ask verification queries where it sets individual rows of \mathbf{R} to zero until verification fails: If the last row set to zero was the i th, then the i th bit of $f_{\mathbf{s}}(m)$ must be 1.¹ Our solution is to randomize the mapping f , i.e., use $f_{\mathbf{s}}(m, \mathbf{b})$ for some randomness \mathbf{b} and compute the tag as $\phi = \pi(\mathbf{R}, \mathbf{R}^\top \cdot f_{\mathbf{s}}(m, \mathbf{b}) + \mathbf{e}, \mathbf{b})$, where π is a pairwise independent permutation (contained in the secret key). We can prove that if LPN is hard then this construction yields a secure MAC. (The key argument is that, with high probability, all non-trivial verification queries are inconsistent and hence lead to reject.) However, the security reduction to the LPN problem is quite loose, concretely, assuming the LPN problem is ϵ hard and the adversary makes Q queries we get security of roughly $Q\sqrt{\epsilon}$. Moreover, there is a non-uniformity issue as we need to know ϵ , or at least have a good guess on it, to achieve this bound (concretely, the optimal choice of parameter μ used in the protocol depends on ϵ , cf. Corollary 4.3).

We also give another construction with a much tighter reduction and completely uniform reduction achieving security roughly $Q\epsilon$. For this, we instantiate the above MAC with a different secret key derivation function $f_{\mathbf{s}}(m, \mathbf{b}) = \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$ (where $\mathbf{v} = h(m, \mathbf{b})$ and $h(\cdot)$ is a pairwise independent hash). The drawback of our second construction is the larger key size, as the secret key contains a matrix $\mathbf{S} \in \mathbb{Z}_2^{\ell \times \mu}$. The security reduction uses a technique from [4, 13] that we needed to adapt to the case of LPN.²

¹While we believe the above scheme would be secure in case verification queries are not allowed, this is of limited interest in practice. Furthermore, the main technical difficulty in building an efficient MAC from LPN seems to be ensuring the secret \mathbf{s} does not leak from verification queries.

²An earlier version of this paper adapted a technique originally used by Waters [44] in the context of IBE schemes that has been applied to lattice-based signature [10] and encryption schemes [1].

Table 1. A comparison of our new authentication protocols with the HB, HB⁺ protocols and the classical GGM construction.

Construction	Achieved security notion	Rounds	Complexity		Key size	Security
			Communication	Computation		
HB [27]	Passive	2	$\ell \cdot n/c$	$\Theta(\ell \cdot n)$	$\ell \cdot c$	ε (tight)
HB ⁺ [28]	Active	3	$\ell \cdot n \cdot 2/c$	$\Theta(\ell \cdot n)$	$\ell \cdot 2 \cdot c$	$\sqrt{\varepsilon}$
AUTH Sect. 3	Active	2	$\ell' \cdot n \cdot (2 + \gamma)/c$	$\Theta(\ell' \cdot n)$	$\ell' \cdot 2(2 + \gamma) \cdot c$	ε (tight)
MAC ₁ Sect. 4.1	MIM from MAC	2	$\ell' \cdot n \cdot (2 + \gamma)/c$	$\Theta(\ell' \cdot n) + \text{PIP}$	$\ell' \cdot 6(2 + \gamma) \cdot c$	$\sqrt{\varepsilon} \cdot Q$ (*)
MAC ₂ Sect. 4.2	MIM from MAC	2	$\ell \cdot n/c$	$\Theta(\ell \cdot n) + \text{PIP}$	$\ell \cdot \lambda \cdot c$	$\varepsilon \cdot Q$
GGM [23]	MIM from PRF	2	$3 \cdot \lambda$	$\Theta(\ell^2 \cdot \lambda)$	$\Theta(\ell)$	$\varepsilon \cdot Q \cdot \lambda$

The trade-off parameter c ($1 \leq c \leq n$) and the term PIP are explained in the ‘‘Communication versus key size’’ paragraph below. The value ε refers to the hardness of the LPN _{τ, ℓ} or SLPN _{τ, L, ℓ'} ^{*} problem. In the last (security) column, we ignore additive terms of the form $Q \cdot 2^{-\Theta(n)}$. (*) is a non-uniform reduction, see discussion in Sect. 4

1.2. Efficiency

Table 1 shows a rough comparison of our new authentication protocols with the HB, HB⁺ protocols and, as a reference, also the protocol one gets by using the PRF constructed via the classical tree-based GGM construction [23] when the underlying PRG is instantiated with LPN. λ denotes a statistical security parameter, say $\lambda = 80$. Q denotes the total number of queries made during the attack. The second row in the table specifies the security notion that is (provably) achieved under the LPN _{τ, ℓ} or ‘‘subset LPN’’ SLPN _{τ, L, ℓ'} ^{*} assumption (all notions will be defined in Sect. 2.4). Typical parameters suggested for HB type protocols [34] are $\ell = 768$, $\tau = 0.05$, at this level they estimate $\varepsilon \approx 2^{-80}$ security for LPN _{τ, ℓ} , which is sufficient for schemes with a tight security reduction like for passive security of HB or active security of our AUTH protocol (cf. the last column in Table 1). The SLPN _{τ, L, ℓ'} ^{*} assumption reduces to LPN _{$\tau, \ell' - g$} with an additive loss of $Q/2^g$, so we should set $\ell' \approx \ell + 80$, say $\ell' = 850$.

The reduction for all these protocols (except GGM) loses an additional additive $Q \cdot 2^{-\Theta(n)}$ term, where n is the number of repetitions required to amplify the security of a basic protocol which only achieves constant soundness and security. The hidden constant in $\Theta(\cdot)$ depends on τ and, for AUTH and MAC₁, on another parameter $\gamma > 0$. An upper bound on the $2^{-\Theta(n)}$ term for AUTH is $2e^{-2(\frac{1}{2} - \frac{1}{2+\gamma})^2 \ell} + 2e^{-2(\frac{1}{4} - \frac{\tau}{2})^2 n}$ (cf. Sect. 3.2), to make this term $< 2^{-80}$ it suffices to set $n = 360$ and $\gamma = 0.75$. Summing up, with $\ell' = 850$, $\tau = 0.05$, $n = 360$, $\gamma = 0.75$ we provably get around 80 bits security for AUTH assuming LPN provides 80 bits security for $\ell = 786$, $\tau = 0.05$.

Computation complexity counts the number of binary operations over \mathbb{F}_2 . Communication complexity counts the total length of all exchanged messages.³ The last row in the

³For MAC- and PRF-based constructions, we consider the communication one incurs by constructing a MIM secure two-round protocol from the MAC (or PRF) by having the prover compute the MAC (PRF output) on a random ρ -bit challenge message. If the adversary can observe Q executions and make q attempts

table states the tightness of the security reduction, i.e., what exact security is achieved (ignoring constants and higher order terms) assuming the $\text{LPN}_{\tau,\ell}$ problem is ε -hard.

The prover and verifier in the HB , HB^+ and our new protocols have to perform $\Theta(\ell \cdot n)$ basic binary operations, assuming the $\text{LPN}_{\tau,\ell}$ problem (i.e., LPN with secrets of length ℓ) is hard. This seems optimal, as $\Theta(\ell)$ operations are necessary to compute the inner product which generates a single pseudorandom bit. We will thus consider an authentication protocol or MAC *efficient*, if it requires $O(\ell \cdot n)$ binary operations. Let us mention that one gets a length-doubling PRG under the $\text{LPN}_{\tau,\ell}$ assumption with $\Theta(\ell^2)$ binary operations [17]. Via the classical GGM construction [23], we obtain a PRF and hence a MAC. This PRF, however, requires $\Theta(\ell^2 \cdot \lambda)$ operations per invocation (where λ is the size of the domain of the PRF) which is not very practical.

Communication Versus Key Size. For all constructions except GGM, there is a natural trade-off between communication and key size, where for any constant c ($1 \leq c \leq n$), we can decrease communication by a factor of c and increase key size by the factor c (cf. Appendix A for how exactly this can be done). For the first three protocols in the table, the choice of c does not affect the computational efficiency, but it does so for the MAC based construction: To compute or verify a tag one has to evaluate a pairwise independent permutation (PIP) on the entire tag of length $l := \Theta(\ell \cdot n/c)$.

There are two standard ways how to construct a PIP π over \mathbb{F}_{2^l} , which we briefly review below together with their corresponding computational complexity:

- Define $\pi(x) := a \cdot x + b \in \mathbb{F}_{2^l}$ for random $a, b \in \mathbb{F}_{2^l}$. Thus the computational cost of evaluating the PIP is one multiplication of two m bits values: Asymptotically, such a multiplication takes only $O(l \log l \log \log l)$ time [18,42], but for small l (like in our scheme) this will not be faster than using schoolbook multiplication, which takes $\Theta(l^2)$ time.
- Define $\pi(\mathbf{x}) := \mathbf{A} \cdot \mathbf{x} + \mathbf{b} \in \{0, 1\}^l$, where we now interpret the input as an l -bit vector, and where \mathbf{A} is a Toeplitz matrix⁴ of dimension l and \mathbf{b} is a random l -bit vector. The product of any Toeplitz matrix with any vector takes time $O(l \log l)$ [2,29], but note that the description of π now requires to store roughly $3l$ bits (whereas the previous solution just requires $2l$ bits).

The PIP term in the table accounts for the above complexity. For parameters $\ell = 850, n = 360, \gamma = 0.75$ as above and trade-off $c = n$ (which minimizes the tag length l) we get $l \approx 2400$ ($2(1 + \gamma)\ell$ plus a statistical security parameter) for MAC_1 and $l \approx 930$ for MAC_2 (ℓ plus a statistical security parameter). Hence, depending on the parameters, the evaluation of the PIP may be the computational bottleneck of our MACs.

Footnote 3 continued

to convince a verifier, the probability of a successful break is at most $Q \cdot q/2^\rho$ (plus the security of the MAC/PRF). As we define security for $q = 1$, setting $\rho := \lambda$ for our statistical security parameter $\lambda = 80$ is sufficient. For the PRF-based construction an output length of λ is sufficient.

⁴Such a matrix is constructed by picking the entries of the first row and column uniformly at random from $\{0, 1\}$, and then copying each value all along its corresponding diagonal.

1.3. Subsequent Work

An abridged version of this paper appeared as [33]. Our results inspired several follow-up works on the problem of efficient authentication from hard learning problems. In [15], Dodis et al. started a systematic study of randomized MACs and showed how to replace the PIP from our MAC constructions with a pairwise independent hash function (leading to more efficient schemes). Heyse et al. [25] considered a variant of our actively secure authentication protocol based on the Ring-LPN problem (introduced in [36]) which simultaneously achieves a very short key and low communication complexity. The hardware efficiency and side-channel security of Lapin was analyzed in [19].

Lyubashevsky and Masny [35] built three-round authentication protocols from LPN that are man-in-the-middle secure for sequential sessions. In [12], Cash et al. show how to reduce the round complexity to two rounds via a generic transformation from active security to sequential man-in-the-middle security.

2. Definitions

2.1. Notation

For a positive integer k , $[k]$ denotes the set $\{1, \dots, k\}$; $[0]$ is the empty set. For $a, b \in \mathbb{R}$, $]a, b[= \{x \in \mathbb{R} ; a < x < b\}$. We denote the set of integers modulo an integer $q \geq 1$ by \mathbb{Z}_q . We will use normal, bold and capital bold letters like x , \mathbf{x} , \mathbf{X} to denote single elements, (column) vectors and matrices over \mathbb{Z}_q , respectively. $\mathbf{X}[i]$ denotes the i th column vector of matrix \mathbf{X} . $\mathbf{x}[i]$ denotes the i th element of vector \mathbf{x} .

For a vector $\mathbf{x} \in \mathbb{Z}_q^m$, $|\mathbf{x}| = m$ denotes the length of \mathbf{x} ; $\mathbf{wt}(\mathbf{x})$ denotes the Hamming weight of the vector \mathbf{x} , i.e., the number of indices $i \in \{1, \dots, |\mathbf{x}|\}$ where $\mathbf{x}[i] \neq 0$. For $\mathbf{v} \in \mathbb{Z}_2^m$ we denote by $\bar{\mathbf{v}}$ its inverse, i.e., $\bar{\mathbf{v}}[i] = 1 - \mathbf{v}[i]$ for all i . For two vectors $\mathbf{v} \in \mathbb{Z}_2^\ell$ and $\mathbf{x} \in \mathbb{Z}_q^\ell$, we denote by $\mathbf{x}_{\downarrow \mathbf{v}} \in \mathbb{Z}_q^{\mathbf{wt}(\mathbf{v})}$ the vector (of length $\mathbf{wt}(\mathbf{v})$) which is derived from \mathbf{x} by deleting all the bits $\mathbf{x}[i]$ where $\mathbf{v}[i] = 0$. If $\mathbf{X} \in \mathbb{Z}_2^{\ell \times m}$ is a matrix, then $\mathbf{X}_{\downarrow \mathbf{v}}$ denotes the sub-matrix obtained by deleting the i th row if $\mathbf{v}[i] = 0$. We also extend Boolean operators to vectors, i.e., for two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_2^m$ we define $\mathbf{x} \wedge \mathbf{y} = \mathbf{z} \in \mathbb{Z}_2^m$ with $\mathbf{z}[i] = \mathbf{x}[i] \wedge \mathbf{y}[i]$ and $\mathbf{x} \vee \mathbf{y} = \mathbf{z} \in \mathbb{Z}_2^m$ where $\mathbf{z}[i] = \mathbf{x}[i] \vee \mathbf{y}[i]$.

A function in λ is *negligible*, written $\text{negl}(\lambda)$, if it vanishes faster than the inverse of any polynomial in λ . An algorithm \mathcal{A} is *probabilistic polynomial time* (PPT) if \mathcal{A} uses some randomness as part of its logic (i.e., \mathcal{A} is probabilistic) and for any input $\mathbf{x} \in \{0, 1\}^*$ the computation of $\mathcal{A}(\mathbf{x})$ terminates in at most $\text{poly}(|\mathbf{x}|)$ steps.

2.2. Authentication Protocols

An authentication protocol is an interactive protocol executed between a prover \mathcal{P} and a verifier \mathcal{V} , both PPT algorithms. Both hold a secret \mathbf{x} (generated using a key generation algorithm \mathcal{K} executed on the security parameter λ in unary) that has been shared in an initial phase. After the execution of the authentication protocol, \mathcal{V} outputs either **accept** or **reject**. We say that the protocol has completeness error α if for all secret keys \mathbf{x} generated by $\mathcal{K}(1^\lambda)$, the honestly executed protocol returns **reject** with probability at most α .

Passive Attacks. An authentication protocol is secure against *passive* attacks if there exists no PPT adversary \mathcal{A} who can win the following game with non-negligible probability: In a first phase, we sample a key $\mathbf{x} \leftarrow \mathcal{K}(1^\lambda)$, and then \mathcal{A} gets to see any number of transcripts from the protocol execution between $\mathcal{P}(\mathbf{x})$ and $\mathcal{V}(\mathbf{x})$ (including \mathcal{V} 's final decision `accept` or `reject`). In a second phase, \mathcal{A} interacts with $\mathcal{V}(\mathbf{x})$ and wins if the verifier outputs `accept`. Here, we only give the adversary one shot to convince the verifier.⁵

Active Attacks. A stronger notion for authentication protocols is security against *active* attacks. Here, the second phase is the same as in a passive attack, but in the first phase, the adversary \mathcal{A} is additionally given access to $\mathcal{P}(\mathbf{x})$. For two-round protocols, there is no difference between concurrent and sequential execution of the sessions with the prover.

We say an authentication protocol is (t, Q, ε) -secure against active adversaries if every adversary \mathcal{A} , running in time at most t and making Q queries to the honest prover, has probability at most ε to win the above game.

Man-in-the-Middle Attacks. The strongest standard security notion for authentication protocols is security against man-in-the-middle (MIM) attacks. Here, the adversary can interact concurrently in many sessions between the prover and the verifier. The adversary gets to learn the verifier `accept/reject` decisions, and wins whenever it manages to let the verifier `accept` in a session where it changed at least one of the messages sent by the prover or the verifier. One can construct two-round authentication schemes which are secure against MIM attacks from basic cryptographic primitives like MACs, which we define next.

2.3. Message Authentication Codes

A message authentication code $\text{MAC} = \{\mathcal{K}, \mathcal{T}, \mathcal{V}\}$ is a triple of algorithms with associated key space \mathcal{K} , message space \mathcal{M} and tag space \mathcal{T} .

- **Key Generation.** The probabilistic key generation algorithm \mathcal{K} takes as input a security parameter $\lambda \in \mathbb{N}$ (in unary) and outputs a secret key $K \in \mathcal{K}$.
- **Tagging.** The probabilistic authentication algorithm \mathcal{T} takes as input a secret key $K \in \mathcal{K}$ and a message $m \in \mathcal{M}$ and outputs an authentication tag $\phi \in \mathcal{T}$.
- **Verification.** The deterministic verification algorithm \mathcal{V} takes as input a secret key $K \in \mathcal{K}$, a message $m \in \mathcal{M}$ and a tag $\phi \in \mathcal{T}$ and outputs `{accept, reject}`.

If the \mathcal{T} algorithm is deterministic, one does not have to explicitly define \mathcal{V} , since it is already defined by the \mathcal{T} algorithm as $\mathcal{V}(K, m, \phi) = \text{accept}$ iff $\mathcal{T}(K, m) = \phi$.

Completeness. We say that MAC has completeness error $\alpha = \alpha(\lambda)$ if for all $m \in \mathcal{M}$ and $\lambda \in \mathbb{N}$

$$\Pr[\mathcal{V}(K, m, \phi) = \text{reject} : K \leftarrow \mathcal{K}(1^\lambda); \phi \leftarrow \mathcal{T}(K, m)] \leq \alpha.$$

⁵By using a hybrid argument, one can show that this implies security even if the adversary can interact in $k \geq 1$ independent instances concurrently (and wins if the verifier accepts in at least one instance). The use of the hybrid argument loses a factor of k in the security reduction.

$\mathbf{Exp}_{\text{MAC}}^{\text{uf-cma}}(\mathcal{A}, \lambda)$ $\begin{array}{l} \mathcal{Q}_{\text{tag}}, \mathcal{Q}_{\text{vrfy}} := \emptyset \\ K \leftarrow \mathcal{K}(1^\lambda) \\ \mathcal{A}^{\text{TAG}(\cdot), \text{VER}(\cdot, \cdot)}(1^\lambda) \\ \text{Return } \left(\begin{array}{l} \exists(m, \phi) \in \mathcal{Q}_{\text{vrfy}} \text{ s.t. } m \notin \mathcal{Q}_{\text{tag}} \\ \wedge \mathcal{V}(K, m, \phi) = \text{accept} \end{array} \right) \end{array}$	<table border="0" style="width: 100%; border-collapse: collapse;"> <tr> <td style="border-bottom: 1px solid black; padding-bottom: 5px;">Oracle TAG(m)</td> </tr> <tr> <td style="padding: 5px;">$\mathcal{Q}_{\text{tag}} := \mathcal{Q}_{\text{tag}} \cup \{m\}$</td> </tr> <tr> <td style="padding: 5px;">$\phi \leftarrow \mathcal{T}(K, m)$</td> </tr> <tr> <td style="padding: 5px;">Return ϕ</td> </tr> <tr> <td style="border-bottom: 1px solid black; padding-bottom: 5px;">Oracle VER(m, ϕ)</td> </tr> <tr> <td style="padding: 5px;">$\mathcal{Q}_{\text{vrfy}} := \mathcal{Q}_{\text{vrfy}} \cup \{(m, \phi)\}$</td> </tr> <tr> <td style="padding: 5px;">Return $\mathcal{V}(K, m, \phi)$</td> </tr> </table>	Oracle TAG(m)	$\mathcal{Q}_{\text{tag}} := \mathcal{Q}_{\text{tag}} \cup \{m\}$	$\phi \leftarrow \mathcal{T}(K, m)$	Return ϕ	Oracle VER(m, ϕ)	$\mathcal{Q}_{\text{vrfy}} := \mathcal{Q}_{\text{vrfy}} \cup \{(m, \phi)\}$	Return $\mathcal{V}(K, m, \phi)$
Oracle TAG(m)								
$\mathcal{Q}_{\text{tag}} := \mathcal{Q}_{\text{tag}} \cup \{m\}$								
$\phi \leftarrow \mathcal{T}(K, m)$								
Return ϕ								
Oracle VER(m, ϕ)								
$\mathcal{Q}_{\text{vrfy}} := \mathcal{Q}_{\text{vrfy}} \cup \{(m, \phi)\}$								
Return $\mathcal{V}(K, m, \phi)$								

Fig. 3. Experiment $\mathbf{Exp}_{\text{MAC}}^{\text{uf-cma}}(\mathcal{A}, \lambda)$ defining uf-cma security of MAC.

Security. The standard security notion for a MAC is unforgeability under a chosen message attack (uf-cma). Formally, this is the probability that the experiment $\mathbf{Exp}_{\text{MAC}}^{\text{uf-cma}}(\mathcal{A}, \lambda)$ of Fig. 3 outputs 1. The experiment features an adversary \mathcal{A} that issues tag queries on messages m , and verification queries on pairs (m, ϕ) ; the adversary is successful if she ever asks a verification query (m, ϕ) that is accepted, for some message m not previously asked to the tag oracle (i.e., \mathcal{A} has found a valid forgery). We say that MAC is (t, Q, ε) -secure against uf-cma adversaries if for any \mathcal{A} running in time t , and asking a total number of Q queries to her oracles, we have $\Pr[\mathbf{Exp}_{\text{MAC}}^{\text{uf-cma}}(\mathcal{A}, \lambda) = 1] \leq \varepsilon$.

2.4. Hard Learning Problems

Let Ber_τ be the Bernoulli distribution over \mathbb{Z}_2 with parameter (bias) $\tau \in]0, 1/2[$ (i.e., $\Pr[x = 1] = \tau$ if $x \leftarrow \text{Ber}_\tau$). For $\ell \geq 1$, Ber_τ^ℓ denotes the distribution over \mathbb{Z}_2^ℓ where each vector consists of ℓ independent samples drawn from Ber_τ . Given a secret $\mathbf{x} \in \mathbb{Z}_2^\ell$ and $\tau \in]0, \frac{1}{2}[$, we write $\Lambda_{\tau, \ell}(\mathbf{x})$ for the distribution over $\mathbb{Z}_2^\ell \times \mathbb{Z}_2$ whose samples are obtained by sampling vectors $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_2^\ell$ and $e \xleftarrow{\$} \text{Ber}_\tau$ outputting $(\mathbf{r}, \mathbf{r}^\top \cdot \mathbf{x} + e)$.

2.4.1. Learning Parity with Noise

The LPN assumption, formally defined below, states that it is hard to distinguish $\Lambda_{\tau, \ell}(\mathbf{x})$ (with a random secret $\mathbf{x} \in \mathbb{Z}_2^\ell$) from the uniform distribution on $\ell + 1$ bits denoted $U_{\ell+1}$.

Definition 2.1. (*Learning Parity with Noise*) The (decisional) $\text{LPN}_{\tau, \ell}$ problem is (t, Q, ε) -hard if for every distinguisher \mathcal{D} running in time t and making Q queries,

$$\left| \Pr \left[\mathcal{D}^{\Lambda_{\tau, \ell}(\mathbf{x})} = 1; \mathbf{x} \xleftarrow{\$} \mathbb{Z}_2^\ell \right] - \Pr \left[\mathcal{D}^{U_{\ell+1}} = 1 \right] \right| \leq \varepsilon.$$

It will sometimes be convenient to think of $U_{\ell+1}$ as LPN samples with uniform errors, note that for any \mathbf{x} , the distributions $\Lambda_{1/2, \ell}(\mathbf{x})$ and $U_{\ell+1}$ are the same.

2.4.2. Subspace Learning Parity with Noise

We now define the (seemingly) stronger *subspace* LPN assumption (SLPN for short) introduced in [40]. Here, the adversary can ask for inner products not only with the

secret \mathbf{x} , but with any affine function $\mathbf{Ax} + \mathbf{b}$ of \mathbf{x} where \mathbf{A} can be any (adversarially and adaptively chosen) matrix of sufficiently high rank. For minimal dimension $d \leq \ell$, a secret $\mathbf{x} \in \mathbb{Z}_2^\ell$ and any $\mathbf{A} \in \mathbb{Z}_2^{\ell \times \ell}$, $\mathbf{b} \in \mathbb{Z}_2^\ell$ we define the distribution

$$\Gamma_{\tau,\ell,d}(\mathbf{x}, \mathbf{A}, \mathbf{b}) = \begin{cases} \perp & \text{if } \text{rank}(\mathbf{A}) < d \\ \Lambda_{\tau,\ell}(\mathbf{A} \cdot \mathbf{x} + \mathbf{b}) & \text{otherwise} \end{cases}$$

and let $\Gamma_{\tau,\ell,d}(\mathbf{x}, \cdot, \cdot)$ denote the oracle which on input \mathbf{A}, \mathbf{b} outputs a sample from $\Gamma_{\tau,\ell,d}(\mathbf{x}, \mathbf{A}, \mathbf{b})$.

Definition 2.2. (*Subspace LPN*) Let $\ell, d \in \mathbb{Z}$ where $d \leq \ell$. The (decisional) $\text{SLPN}_{\tau,\ell,d}$ problem is (t, Q, ε) -hard if for every distinguisher \mathcal{D} running in time t and making Q queries,

$$\left| \Pr \left[\mathcal{D}^{\Gamma_{\tau,\ell,d}(\mathbf{x}, \cdot, \cdot)} = 1; \mathbf{x} \xleftarrow{\$} \mathbb{Z}_2^\ell \right] - \Pr \left[\mathcal{D}^{U_{\ell+1}(\cdot, \cdot)} = 1 \right] \right| \leq \varepsilon,$$

where $U_{\ell+1}(\cdot, \cdot)$ on input \mathbf{A}, \mathbf{b} outputs a sample of $U_{\ell+1}$ if $\text{rank}(\mathbf{A}) \geq d$ and \perp otherwise.

The following proposition states that the subspace LPN problem mapping to dimension $d + g$ is almost as hard as the standard LPN problem with secrets of length d , the hardness gap being exponentially small in g . This proposition is a special case of Theorem 1 from [40]. To be self-contained, we include a proof of this proposition, but first we state a much simpler result, which will be used in the construction of our second MAC, namely that the subspace LPN problem is exactly as hard as the LPN problem if $d = \ell$, i.e., the subspace must span the entire space.

Proposition 2.3. *For any $\ell \in \mathbb{Z}$ if the $\text{LPN}_{\tau,\ell}$ problem is (t, Q, ε) -hard then the $\text{SLPN}_{\tau,\ell,\ell}$ problem is (t', Q, ε) -hard where $t' = t - \text{poly}(\ell, Q)$*

Proof. Given a $\Lambda_{\tau,\ell}(\mathbf{x})$ sample $(\mathbf{r}, \mathbf{r}^\top \mathbf{x} + e)$, for any \mathbf{A}, \mathbf{b} where $\text{rank}(\mathbf{A}) = \ell$, we can transform it into a $\Gamma_{\tau,\ell,\ell}(\mathbf{x}, \mathbf{A}, \mathbf{b})$ sample $(\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top (\mathbf{Ax} + \mathbf{b}) + e)$ by setting $\hat{\mathbf{r}}^\top = \mathbf{r}^\top \mathbf{A}^{-1}$ and outputting $(\hat{\mathbf{r}}, \mathbf{r}^\top \mathbf{x} + e + \hat{\mathbf{r}}^\top \mathbf{b})$. We now have

$$(\hat{\mathbf{r}}, \mathbf{r}^\top \mathbf{x} + e + \hat{\mathbf{r}}^\top \mathbf{b}) = (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{Ax} + e + \hat{\mathbf{r}}^\top \mathbf{b}) = (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top (\mathbf{Ax} + \mathbf{b}) + e)$$

It just remains to check that e and $\hat{\mathbf{r}}$ have the right distribution. The error e in the subspace LPN sample is exactly the same value as in the LPN sample and thus also distributed as Ber_τ . $\hat{\mathbf{r}}^\top$ is uniform as required, to see this recall that \mathbf{r} is uniform and \mathbf{A} has full rank. Using this transformation, we can use any distinguisher for $\text{SLPN}_{\tau,\ell,\ell}$ from uniform to distinguish $\text{LPN}_{\tau,\ell}$ from uniform with the same advantage. \square

Proposition 2.4. *For any $\ell, d, g \in \mathbb{Z}$ (where $\ell \geq d + g$), if the $\text{LPN}_{\tau,d}$ problem is (t, Q, ε) -hard then the $\text{SLPN}_{\tau,\ell,d+g}$ problem is (t', Q, ε') -hard where*

$$t' = t - \text{poly}(\ell, Q) \quad \varepsilon' = \varepsilon + Q/2^g.$$

Proof. Let \mathcal{D}' be an adversary with advantage ε' for the $\text{SLPN}_{\tau, \ell, d+g}$ problem, from this \mathcal{D}' we will construct an \mathcal{D} with advantage $\varepsilon \geq \varepsilon' - Q/2^g$ for the $\text{LPN}_{\tau, d}$ problem. W.l.o.g., we assume that the oracle queries of \mathcal{D}' are of the form \mathbf{A}, \mathbf{b} with $\mathbf{b} = \mathbf{0}$ since $(\hat{\mathbf{r}}, \hat{\mathbf{z}}) \mapsto (\hat{\mathbf{r}}, \hat{\mathbf{z}} + \hat{\mathbf{r}}^\top \mathbf{b})$ reduces queries with arbitrary \mathbf{b} to queries with $\mathbf{b} = \mathbf{0}$.

Our \mathcal{D} will transform the samples of the form $(\mathbf{r}, \mathbf{r}^\top \mathbf{x} + e)$ it gets (where e is either sampled according to Ber_τ or uniform) into samples $(\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{A} \hat{\mathbf{x}} + e)$ for any $\mathbf{A} \in \mathbb{Z}_2^{\ell \times \ell}$ of rank $\geq d + g$. In particular, LPN samples are mapped to SLPN samples, and random samples are mapped to random samples. For each of the Q queries made by \mathcal{D}' , the transformation will fail with probability at most 2^{-g} , which is where the $Q/2^g$ loss in distinguishing advantage comes from. We now formally define \mathcal{D} .

Initially, $\mathcal{D}^{\Lambda_{\delta, d}(\mathbf{x})}$ (where δ is either τ or $1/2$) samples $\mathbf{W} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times d}$, $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_2^\ell$ which (implicitly) defines the secret $\hat{\mathbf{x}} = \mathbf{W}\mathbf{x} + \mathbf{w}$ for the transformation. Now, $\mathcal{D}^{\Lambda_{\delta, d}(\mathbf{x})}$ invokes \mathcal{D}' and answers every query \mathbf{A} of \mathcal{D}' as follows.

If $\text{rank}(\mathbf{A}) < d + g$ return \perp to \mathcal{D}' . Otherwise, query the oracle $\Lambda_{\delta, d}(\mathbf{x})$ to get a sample $(\mathbf{r}, \mathbf{r}^\top \cdot \mathbf{x} + e)$. Define the set $\mathcal{S} \subseteq \mathbb{Z}_2^\ell$ of solutions to the system of linear equations:

$$\mathcal{S} = \left\{ \mathbf{y} : \mathbf{y}^\top \mathbf{A} \mathbf{W} = \mathbf{r}^\top \right\} \subseteq \mathbb{Z}_2^\ell$$

Note that if $\mathbf{A} \mathbf{W}$ has rank d then \mathcal{S} is non-empty. \mathcal{D} samples $\hat{\mathbf{r}} \xleftarrow{\$} \mathcal{S}$ and outputs the sample

$$(\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{x} + \hat{\mathbf{r}}^\top \mathbf{A} \mathbf{w} + e), \quad (2.1)$$

Note that \mathcal{D} runs in time t' (as it must invoke \mathcal{D}') plus some $\text{poly}(\ell, Q)$ overhead as claimed. It remains to show that simulation performed by \mathcal{D} is correct. This is shown in the following claims.

Claim 2.5. *If $\mathbf{V} = \mathbf{A} \mathbf{W}$ has (full) rank d , then $\hat{\mathbf{r}} \xleftarrow{\$} \mathcal{S}$ is uniformly random (given $\mathbf{A}, \mathbf{W}, \mathbf{w}$).*

Proof of Claim. We show that for any $\mathbf{v} \in \mathbb{Z}_2^\ell$, $\Pr[\hat{\mathbf{r}} = \mathbf{v} \mid \mathbf{W}, \mathbf{A}, \mathbf{w}] = 2^{-\ell}$. First, as $\mathbf{r} \in \mathbb{Z}_2^d$ is uniform, $\Pr[\mathbf{v}^\top \mathbf{A} \mathbf{W} = \mathbf{r}^\top] = 2^{-d}$, if this does not hold, then $\mathbf{v} \neq \hat{\mathbf{r}}$. Otherwise, $\hat{\mathbf{r}}$ is sampled at uniform from an $\ell - d$ dimensional linear space, and thus $\Pr[\mathbf{v} = \hat{\mathbf{r}} \mid \mathbf{v}^\top \mathbf{A} \mathbf{W} = \mathbf{r}^\top] = 2^{d-\ell}$. We get

$$\Pr[\hat{\mathbf{r}} = \mathbf{v} \mid \mathbf{W}, \mathbf{A}, \mathbf{w}] = \Pr[\mathbf{v}^\top \mathbf{A} \mathbf{W} = \mathbf{r}^\top] \Pr[\mathbf{v} = \hat{\mathbf{r}} \mid \mathbf{v}^\top \mathbf{A} \mathbf{W} = \mathbf{r}^\top] = 2^{-d} 2^{d-\ell} = 2^{-\ell}. \quad \square$$

Claim 2.6. *\mathcal{D} perfectly simulates the distribution $\Gamma_{\delta, \ell, d+g}(\hat{\mathbf{x}}, \mathbf{A})$ (where $\hat{\mathbf{x}} = \mathbf{W}\mathbf{x} + \mathbf{w}$).*

Proof of Claim. We can rewrite the samples of Eq. (2.1) as

$$\begin{aligned} (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{x} + \hat{\mathbf{r}}^\top \mathbf{A} \mathbf{w} + e) &= (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{A} \mathbf{W} \mathbf{x} + \hat{\mathbf{r}}^\top \mathbf{A} \mathbf{w} + e) \quad (\text{since } \hat{\mathbf{r}} \in \mathcal{S}) \\ &= (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{A} (\mathbf{W} \mathbf{x} + \mathbf{w}) + e) \\ &= (\hat{\mathbf{r}}, \hat{\mathbf{r}}^\top \mathbf{A} \hat{\mathbf{x}} + e) \end{aligned}$$

which is a sample from $\Gamma_{\delta,\ell,d+g}(\hat{\mathbf{x}}, \mathbf{A})$ as required. \square

Claim 2.7. *The probability that the set \mathcal{S} is empty is at most 2^{-g} .*

Proof of Claim. Recall that the set \mathcal{S} can possibly be empty only when $\mathbf{V} = \mathbf{A}\mathbf{W} \in \mathbb{Z}_2^{\ell \times d}$ has rank less than d , where $\mathbf{A} \in \mathbb{Z}_2^{\ell \times \ell}$ has rank $\geq d + g$ and $\mathbf{W} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times d}$.

Denote with $\Delta(d, g)$ the probability that a random matrix in $\mathbb{Z}_2^{(d+g) \times d}$ has rank less than d . Since the matrix \mathbf{A} has rank at least $d + g$, we can assume, without loss of generality, that the first $d + g$ rows of \mathbf{A} are linearly independent. Since the matrix \mathbf{W} is random, the upper $(d + g) \times d$ matrix of $\mathbf{V} = \mathbf{A}\mathbf{W}$ is random in $\mathbb{Z}_2^{(d+g) \times d}$ and thus it has rank less than d with probability at most $\Delta(d, g)$. We conclude that \mathbf{V} has rank strictly less than d exactly with the same probability. Using Lemma 4.20, we see that this probability is bounded by 2^{-g} .

Applying the union bound, we can upper bound the probability that for any of the Q queries the matrix $\mathbf{V} = \mathbf{A}\mathbf{W}$ has rank less than d by $Q \cdot 2^{-g}$. This error probability is thus an upper bound on the gap of the success probability ε' of \mathcal{D}' and the success probability ε we get in breaking LPN using the transformation.

Finally, we need to consider the fact that the queries \mathbf{A} chosen by \mathcal{D}' are chosen adaptively. To show that adaptivity does not help in picking an \mathbf{A} where $\mathbf{A}\mathbf{W}$ has rank $< d$, we must show that the view of \mathcal{D}' is independent of \mathbf{W} (except for the fact that so far no query was made where $\text{rank}(\mathbf{A}\mathbf{W}) < d$). This holds as the secret $\hat{\mathbf{x}} = \mathbf{W}\mathbf{x} + \mathbf{w}$ used in the simulation is independent of \mathbf{W} as it is blinded with a uniform \mathbf{w} . In fact, the only reason we use this blinding is to enforce this independence. \square

2.4.3. Subset Learning Parity with Noise

For some of our constructions, we will only need a weaker version of the $\text{SLPN}_{\tau,\ell,d}$ problem that we call **subset LPN**. As the name suggests, here the adversary does not ask for inner products with $\mathbf{A}\mathbf{x} + \mathbf{b}$ for any \mathbf{A} (of rank $\geq d$), but only with *subsets* of \mathbf{x} (of size $\geq d$). It will be convenient to explicitly define this special case. For $\mathbf{x}, \mathbf{v} \in \mathbb{Z}_2^\ell$, let $\text{diag}(\mathbf{v}) \in \mathbb{Z}_2^{\ell \times \ell}$ denote the matrix with \mathbf{v} in the diagonal and zero elsewhere, and let

$$\Gamma_{\tau,\ell,d}^*(\mathbf{x}, \mathbf{v}) := \Gamma_{\tau,\ell,d}(\mathbf{x}, \text{diag}(\mathbf{v})) = \begin{cases} \perp & \text{if } \text{wt}(\mathbf{v}) < d \\ \Lambda_{\tau,\ell}(\mathbf{x} \wedge \mathbf{v}) & \text{otherwise.} \end{cases}$$

Definition 2.8. (*Subset LPN*) Let $\ell, d \in \mathbb{Z}$ where $d \leq \ell$. The $\text{SLPN}_{\tau,\ell,d}^*$ problem is (t, Q, ε) -hard if for every distinguisher \mathcal{D} running in time t and making Q queries,

$$\left| \Pr \left[\mathcal{D}^{\Gamma_{\tau,\ell,d}^*(\mathbf{x}, \cdot)} = 1; \mathbf{x} \xleftarrow{\$} \mathbb{Z}_2^\ell \right] - \Pr \left[\mathcal{D}^{U_{\ell+1}(\cdot)} = 1 \right] \right| \leq \varepsilon,$$

where $U_{\ell+1}(\cdot)$ on input \mathbf{v} (where $\text{wt}(\mathbf{v}) \geq d$) outputs a sample of $U_{\ell+1}$ and \perp otherwise.

Remark 2.9. $\Gamma_{\tau,\ell,d}^*(\mathbf{x}, \mathbf{v})$ samples are of the form $(\mathbf{r}, \mathbf{r}_{\downarrow \mathbf{v}}^\top \mathbf{x}_{\downarrow \mathbf{v}} + e) \in \mathbb{Z}_2^{\ell+1}$, where $e \xleftarrow{\$} \text{Ber}_\tau$. To compute the inner product only $\mathbf{r}_{\downarrow \mathbf{v}} \in \mathbb{Z}_2^{\text{wt}(\mathbf{v})}$ is needed, the remaining bits

$\mathbf{r}_{\downarrow \bar{v}} \in \mathbb{Z}_2^{\ell - \text{wt}(\mathbf{v})}$ are irrelevant. We use this observation to improve the communication complexity (for protocols) or tag length (for MACs), by using “compressed” samples of the form $(\mathbf{r}_{\downarrow \mathbf{v}}, \mathbf{r}_{\downarrow \mathbf{v}}^\top \mathbf{x}_{\downarrow \mathbf{v}} + e) \in \mathbb{Z}_2^{\text{wt}(\mathbf{v})+1}$.

2.5. Hoeffding Inequality

We will several times use the following tail inequality known as Hoeffding’s inequality.

Theorem 2.10. [26] *Let X_1, X_2, \dots, X_n be i.i.d. Bernoulli variables with $\Pr[X_i = 1] = p$ and $X = \sum_{i=1}^n X_i$ (so $\mathbb{E}[X] = np$). Then, for $\epsilon \geq 0$,*

$$\Pr[X - np \geq \epsilon n] \leq e^{-2\epsilon^2 n}$$

$$\Pr[|X - np| \geq \epsilon n] \leq 2e^{-2\epsilon^2 n}.$$

The above also holds if the X_i are sampled without replacement.

3. Two-Round Authentication with Active Security

In this section, we describe our new two-round authentication protocol and prove its active security under the hardness of the $\text{SLPN}_{\tau, 2\ell, d}^*$ problem, where $d = \ell/(2 + \gamma)$ for some constant $\gamma > 0$.

- **Public Parameters.** The authentication protocol has the following public parameters, where τ, τ' are constants and ℓ, n depend on the security parameter λ .⁶

$\ell \in \mathbb{N}$	length of the secret key $\mathbf{s} \in \mathbb{Z}_2^{2\ell}$
$\tau \in]0, 1/2[$	parameter of the Bernoulli error distribution Ber_τ
$\tau' = 1/4 + \tau/2$	acceptance threshold
$n \in \mathbb{N}$	number of parallel repetitions (we require $\ell \geq 2n$)

- **Key Generation.** Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_2^{2\ell}$ and returns \mathbf{s} as the secret key.
- **Authentication Protocol.** The two-round authentication protocol with prover $\mathcal{P}_{\tau, n}$ and verifier $\mathcal{V}_{\tau', n}$ is given in Fig. 4.

Theorem 3.1. *For any constant $\gamma > 0$, let $d = \ell/(2 + \gamma)$. If the $\text{SLPN}_{\tau, 2\ell, d}^*$ problem is (t, nQ, ϵ) -hard, then the authentication protocol from Fig. 4 is (t', Q, ϵ') -secure against active adversaries, where for constants $c_\gamma, c_\tau > 0$ that depend only on γ and τ , respectively,*

$$t' = t - \text{poly}(Q, \ell) \quad \epsilon' = \epsilon + Q \cdot 2^{-c_\gamma \cdot \ell} + 2^{-c_\tau \cdot n} = \epsilon + Q \cdot 2^{-\Theta(n)}.$$

⁶The acceptance threshold τ' below can be any value in-between τ and $1/2$. As we set it closer to $1/2$, the soundness error increases while the completeness error decreases. For concreteness, we set τ' exactly in-between τ and $1/2$.

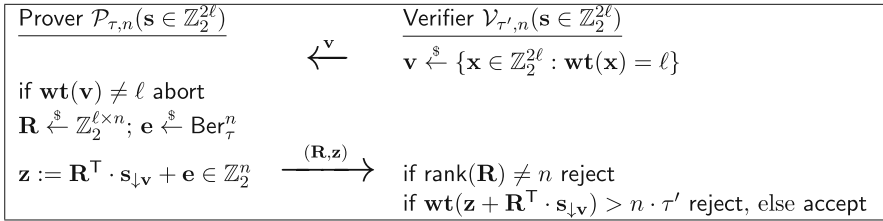


Fig. 4. Two-round protocol AUTH with active security from the LPN assumption.

The protocol has completeness error $2^{-c'_\tau \cdot n} + 2^{n-\ell} \in 2^{-\Theta(n)}$ (as $\ell \geq 2n$) where $c'_\tau > 0$ depends only on τ .

3.1. Proof of Completeness

We first bound the completeness error α of the authentication protocol. For any $n \in \mathbb{N}$, $\tau \in]0, 1/2[$, let

$$\alpha_{\tau,n} := \Pr[\text{wt}(\mathbf{e}) > n \cdot \tau' : \mathbf{e} \xleftarrow{\$} \text{Ber}_{\tau}^n] \stackrel{(\text{Thm.2.10})}{\leq} e^{-2(1/4-\tau/2)^2 n} \stackrel{\text{def}}{=} 2^{-c'_\tau \cdot n} \quad (3.1)$$

denote the probability that n independent Bernoulli samples with bias τ contain more than a $\tau' := 1/4 + \tau/2$ fraction of 1's (i.e., $n(1/4 + \tau/2) - n\tau = n(1/4 - \tau/2)$ more than the expected number).

The verifier performs the following two checks. In the first verification step, the verifier rejects if the random matrix \mathbf{R} does not have full rank. By Lemma 4.20 (cf. Appendix B), the probability of this event is $\leq 2^{n-\ell}$. Now, let $\mathbf{e} := \mathbf{z} + \mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}}$ denote the noise added by $\mathcal{P}_{\tau,n}$. Then, in the second verification step, the verifier rejects if $\text{wt}(\mathbf{e}) > n \cdot \tau'$. From Eq. (3.1), we have that this happens with probability $\alpha_{\tau,n}$. It follows that $\alpha \leq 2^{n-\ell} + 2^{-c'_\tau \cdot n}$.

3.2. Proof of Security

We first define some more terms that will be used later in the security proof. For a constant $\gamma > 0$, let $d = \ell/(2 + \gamma)$ (as in Theorem 3.1). Let $\alpha'_{\ell,d}$ denote the probability that a random substring of length ℓ chosen from a string of length 2ℓ with Hamming weight ℓ , has a Hamming weight less than $d = \ell/(2 + \gamma)$. Using the Hoeffding bound, we see there is a constant $c_\gamma > 0$ s.t.

$$\alpha'_{\ell,d} \stackrel{(\text{Thm.2.10})}{\leq} 2e^{-2\left(\frac{1}{2} - \frac{1}{2+\gamma}\right)^2 \ell} \stackrel{\text{def}}{=} 2^{-c_\gamma \cdot \ell}. \quad (3.2)$$

For $\tau' = 1/4 + \tau/2$, let $\alpha''_{\tau',n}$ denote the probability that a random bitstring $\mathbf{y} \in \mathbb{Z}_2^n$ has Hamming weight $\text{wt}(\mathbf{y}) \leq n \cdot \tau'$. From the Hoeffding bound, it follows that there exists a constant $c_\tau > 0$ (only depending on τ), such that (below we use $1/2 - \tau' = 1/4 - \tau/2$)

$$\alpha''_{\tau',n} := 2^{-n} \cdot \sum_{i=0}^{\lfloor n \cdot \tau' \rfloor} \binom{n}{i} \stackrel{(\text{Thm.2.10})}{\leq} 2e^{-2\left(\frac{1}{4} - \frac{\tau'}{2}\right)^2 n} \stackrel{\text{def}}{=} 2^{-c_{\tau'} \cdot n}. \quad (3.3)$$

We now prove security of the authentication protocol. Consider an oracle \mathcal{O} which is either the subset LPN oracle $\Gamma_{\tau,2\ell,d}^*(\mathbf{x}, \cdot)$ or $U_{2\ell+1}(\cdot)$, as defined in Definition 2.8. We will construct an adversary $\mathcal{B}^{\mathcal{O}}$ that uses \mathcal{A} (who breaks the active security of AUTH with advantage ε') in a black-box way such that:

$$\Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,d}^*} = 1] \geq \varepsilon' \quad \text{and} \quad \Pr[\mathcal{B}^{U_{2\ell+1}} = 1] \leq \alpha''_{\tau',n} + Q \cdot \alpha'_{\ell,d}.$$

Thus $\mathcal{B}^{\mathcal{O}}$ can distinguish between the two oracles with advantage $\varepsilon := \varepsilon' - Q \cdot \alpha'_{\ell,d} - \alpha''_{\tau',n} = \varepsilon + Q \cdot 2^{-c_{\tau'} \cdot \ell} + 2^{-c_{\tau'} \cdot n}$ as claimed in the statement of the theorem. Below we define $\mathcal{B}^{\mathcal{O}}$.

Setup Initially, $\mathcal{B}^{\mathcal{O}}$ samples

$$\mathbf{x}^* \xleftarrow{\$} \mathbb{Z}_2^{2\ell}, \quad \mathbf{v}^* \xleftarrow{\$} \left\{ \mathbf{y} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{y}) = \ell \right\}.$$

The intuition of our simulation below is as follows. Let us first assume \mathcal{O} is a subset LPN oracle $\Gamma_{\tau,2\ell,d}^*(\mathbf{x}, \cdot)$ with secret \mathbf{x} . To simulate the prover during the first phase we have to produce answers $\phi = (\mathbf{R}, \mathbf{z})$ to each query $\mathbf{v} \in \{\mathbf{y} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{y}) = \ell\}$ issued by \mathcal{A} . The simulated answers will have the same distribution as the answers of an honest prover $\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$ where

$$\mathbf{s} = (\mathbf{x}^* \wedge \mathbf{v}^*) + (\mathbf{x} \wedge \bar{\mathbf{v}}^*). \quad (3.4)$$

Thus one half of the bits of \mathbf{s} come from \mathbf{x}^* , and the other half come from the unknown secret \mathbf{x} (for which we use the oracle \mathcal{O}). In the second phase, we give \mathcal{A} the challenge \mathbf{v}^* . As $\mathbf{s}_{\downarrow \mathbf{v}^*} = (\mathbf{x}^* \wedge \mathbf{v}^*)_{\downarrow \mathbf{v}^*} = \mathbf{x}^*_{\downarrow \mathbf{v}^*}$ is known, we will be able to verify if \mathcal{A} did output a valid answer. On the other hand, if \mathcal{O} is the uniform oracle $U_{2\ell+1}(\cdot)$, then after the interaction with the prover we will show that $\mathbf{s}_{\downarrow \mathbf{v}^*} = (\mathbf{x}^* \wedge \mathbf{v}^*)_{\downarrow \mathbf{v}^*}$ is information theoretically hidden, and thus \mathcal{A} cannot succeed except with exponentially small probability.

First phase

In the first phase $\mathcal{B}^{\mathcal{O}}$ invokes \mathcal{A} who expects access to $\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$. We now specify how $\mathcal{B}^{\mathcal{O}}$ samples the answer $\phi = (\mathbf{R}, \mathbf{z})$ to a query $\mathbf{v} \in \{\mathbf{y} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{y}) = \ell\}$ made by \mathcal{A} . Let

$$\mathbf{u}^* := \mathbf{v} \wedge \mathbf{v}^* \quad \mathbf{u} := \mathbf{v} \wedge \bar{\mathbf{v}}^*.$$

1. $\mathcal{B}^{\mathcal{O}}$ queries its oracle n times on the input \mathbf{u} . If the oracle's output is \perp (this is the case if $\mathbf{wt}(\mathbf{u}) < d$), then $\mathcal{B}^{\mathcal{O}}$ outputs 1 and stops. Otherwise, let $\hat{\mathbf{R}}_1 \in \mathbb{Z}_2^{2\ell \times n}$, $\mathbf{z}_1 \in \mathbb{Z}_2^n$ denote the n outputs of the oracle.

2. Sample $\hat{\mathbf{R}}_0 \xleftarrow{\$} \mathbb{Z}_2^{2\ell \times n}$ and set $\mathbf{z}_0 = \hat{\mathbf{R}}_0^\top \cdot (\mathbf{x}^* \wedge \mathbf{u}^*)$.
3. Return $\phi = (\mathbf{R} = \hat{\mathbf{R}}_{\downarrow \mathbf{v}} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} = \mathbf{z}_0 + \mathbf{z}_1 \in \mathbb{Z}_2^n)$, where $\hat{\mathbf{R}}$ is uniquely determined by requiring $\hat{\mathbf{R}}_{\downarrow \mathbf{v}^*} = \hat{\mathbf{R}}_0$ and $\hat{\mathbf{R}}_{\downarrow \bar{\mathbf{v}}^*} = \hat{\mathbf{R}}_1$.

Second phase Eventually, \mathcal{A} enters the second phase of the active attack expecting a challenge from $\mathcal{V}_{\tau',n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$.

1. $\mathcal{B}^{\mathcal{O}}$ forwards \mathbf{v}^* as the challenge to \mathcal{A} .
2. \mathcal{A} answers with some $(\mathbf{R}^*, \mathbf{z}^*)$.
3. $\mathcal{B}^{\mathcal{O}}$ checks if

$$\text{rank}(\mathbf{R}^*) = n \quad \text{and} \quad \text{wt}(\mathbf{z}^* + \mathbf{R}^{*\top} \cdot \mathbf{x}_{\downarrow \mathbf{v}^*}^*) \leq n \cdot \tau'. \tag{3.5}$$

The output is 1 if both checks succeed and 0 otherwise.

Claim 3.2. $\Pr[\mathcal{B}^{U_{2\ell+1}(\cdot)} = 1] \leq \alpha''_{\tau',n} + Q \cdot \alpha'_{\ell,d}$.

Proof of Claim. Recall that we defined \mathcal{B} such that it will output 1 and stop in the first phase of the experiment whenever it receives \perp as output from its oracle. We defined $U_{2\ell+1}(\cdot)$ to output \perp if it is queried on an input $\mathbf{u} = \mathbf{v} \wedge \bar{\mathbf{v}}^*$ where $\text{wt}(\mathbf{u}) < d$. Since $\mathbf{v}^* \xleftarrow{\$} \{\mathbf{y} \in \mathbb{Z}_2^{2\ell} : \text{wt}(\mathbf{y}) = \ell\}$, for any \mathbf{v} , the probability that $\text{wt}(\mathbf{v} \wedge \bar{\mathbf{v}}^*) < d$ is (by definition) $\alpha'_{\ell,d}$ as defined in Eq. (3.2). Using the union bound, we can upper bound the probability that $\text{wt}(\mathbf{v} \wedge \bar{\mathbf{v}}^*) < d$ for any of the Q different \mathbf{v} 's chosen by the adversary in the first phase as $Q \cdot \alpha'_{\ell,d}$.

If we did not output 1 in the first phase and in the second phase \mathbf{R}^* does not have full rank then \mathcal{B} outputs 0 by definition. Therefore, we now consider the case where $\text{rank}(\mathbf{R}^*) = n$.

The answers $\phi = (\mathbf{R}, \mathbf{z})$ that the adversary \mathcal{A} obtained from $\mathcal{B}^{U_{2\ell+1}(\cdot)}$ in the first phase are independent of \mathbf{x}^* (i.e., $\mathbf{z} = \mathbf{z}_0 + \mathbf{z}_1$ is uniform as \mathbf{z}_1 is uniform). Since $\mathbf{x}_{\downarrow \mathbf{v}^*}^*$ is uniformly random and \mathbf{R}^* has full rank, the vector

$$\mathbf{y} := \mathbf{R}^{*\top} \cdot \mathbf{x}_{\downarrow \mathbf{v}^*}^* + \mathbf{z}^*$$

is uniformly random over \mathbb{Z}_2^n . Thus the probability that the second verification in Eq. (3.5) does not fail is $\Pr[\text{wt}(\mathbf{y}) \leq n \cdot \tau'] = \alpha''_{\tau',n}$. □

Claim 3.3. $\Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,d}^*(\mathbf{x},\cdot)} = 1] \geq \varepsilon'$

Proof of Claim. We will show that \mathcal{B} outputs 1 with probability $\geq \varepsilon'$ if the subset LPN oracle accepts subsets of arbitrary small size (and does not simply output \perp on inputs \mathbf{v} where $\text{wt}(\mathbf{v}) < d$), i.e.,

$$\Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,0}^*(\mathbf{x},\cdot)} = 1] \geq \varepsilon'. \tag{3.6}$$

This then implies the claim using

$$\Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,d}^*(\mathbf{x},\cdot)} = 1] \geq \Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,0}^*(\mathbf{x},\cdot)} = 1]$$

which holds as the oracles $\Gamma_{\tau,2\ell,d}^*(\mathbf{x}, \cdot)$ and $\Gamma_{\tau,2\ell,0}^*(\mathbf{x}, \cdot)$ behave the same except on inputs \mathbf{v} with $\text{wt}(\mathbf{v}) < d$ in which case the former outputs \perp , and \mathcal{B} will then immediately output 1.

Equation (3.6) holds as:

- The answers $\phi = (\mathbf{R}, \mathbf{z})$ that $\mathcal{B}^{\Gamma_{\tau,2\ell,0}^*(\mathbf{x}, \cdot)}$ gives to \mathcal{A} 's queries in the first phase of the attack have *exactly* the same distribution as what \mathcal{A} would get when interacting with an honest prover $\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$ where the “simulated” secret \mathbf{s} is defined in Eq. (3.4).

To see this, recall that on a query \mathbf{v} from \mathcal{A} , adversary $\mathcal{B}^{\Gamma_{\tau,2\ell,0}^*(\mathbf{x}, \cdot)}$ must compute n SLPN samples $(\hat{\mathbf{R}}, \mathbf{z} = \hat{\mathbf{R}}^\top \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e})$ and then forward the compressed version of this samples to \mathcal{A} (that is, $(\mathbf{R}, \mathbf{z} = \mathbf{R}^\top \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e})$ where $\mathbf{R} = \hat{\mathbf{R}}_{\downarrow \mathbf{v}}$, cf. Remark 2.9). We next show that the \mathbf{z} computed by \mathcal{B} indeed has exactly this distribution. In the first step, \mathcal{B} queries its oracle with $\mathbf{u} = \mathbf{v} \wedge \bar{\mathbf{v}}^*$ and obtains noisy inner products $(\hat{\mathbf{R}}_1, \mathbf{z}_1)$ with the part of $\mathbf{s}_{\downarrow \mathbf{v}}$ that contains only bits from \mathbf{x} , i.e.,

$$\mathbf{z}_1 = \hat{\mathbf{R}}_1^\top \cdot (\mathbf{x} \wedge \mathbf{u}) + \mathbf{e} = \hat{\mathbf{R}}_1^\top \cdot (\mathbf{s} \wedge \mathbf{u}) + \mathbf{e}.$$

In the second step, \mathcal{B} samples n inner products $(\hat{\mathbf{R}}_0, \mathbf{z}_0)$ (with no noise) with the part of $\mathbf{s}_{\downarrow \mathbf{v}}$ that contains only bits from the known \mathbf{x}^* , i.e.,

$$\mathbf{z}_0 = \hat{\mathbf{R}}_0^\top \cdot (\mathbf{x}^* \wedge \mathbf{u}^*) = \hat{\mathbf{R}}_0^\top \cdot (\mathbf{s} \wedge \mathbf{u}^*).$$

In the third step, \mathcal{B} then generates $(\hat{\mathbf{R}}, \hat{\mathbf{R}}^\top \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e})$ from the previous values where $\hat{\mathbf{R}}$ is defined by $\hat{\mathbf{R}}_{\downarrow \mathbf{v}^*} = \hat{\mathbf{R}}_0$ and $\hat{\mathbf{R}}_{\downarrow \bar{\mathbf{v}}^*} = \hat{\mathbf{R}}_1$. Using $\mathbf{v} = \mathbf{u} + \mathbf{u}^*$, we get

$$\begin{aligned} \mathbf{z} &= \mathbf{z}_0 + \mathbf{z}_1 \\ &= \hat{\mathbf{R}}_0^\top \cdot (\mathbf{s} \wedge \mathbf{u}^*) + \hat{\mathbf{R}}_1^\top \cdot (\mathbf{s} \wedge \mathbf{u}) + \mathbf{e} \\ &= \hat{\mathbf{R}}^\top \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e}. \end{aligned}$$

- The challenge \mathbf{v}^* sent initially to \mathcal{A} is uniformly random and therefore has the same distribution as a challenge in an active attack.
- $\mathcal{B}^{\Gamma_{\tau,2\ell,0}^*(\mathbf{x}, \cdot)}$ outputs 1 if Eq. (3.5) holds, which is exactly the case when \mathcal{A} 's response to the challenge was valid. By assumption this probability is at least ε' .

This concludes the proof of Eq. (3.6). \square

3.3. Avoiding Checking

One disadvantage of the protocol in Fig. 4, compared to HB style protocols, is the necessity to check whether the messages exchanged have the right form: the prover checks if \mathbf{v} has weight ℓ , while the verifier must make the even more expensive check whether \mathbf{R} has full rank. We note that it is possible to eliminate these checks by blinding the exchanged messages \mathbf{v} and \mathbf{z} using random vectors $\mathbf{b}^{\mathbf{v}} \in \mathbb{Z}_2^{2\ell}$ and $\mathbf{b}^{\mathbf{z}} \in \mathbb{Z}_2^n$, respectively, as shown in Fig. 5. The analysis of this protocol is similar as for the protocol in Fig. 5, we thus just sketch a proof emphasizing the differences.

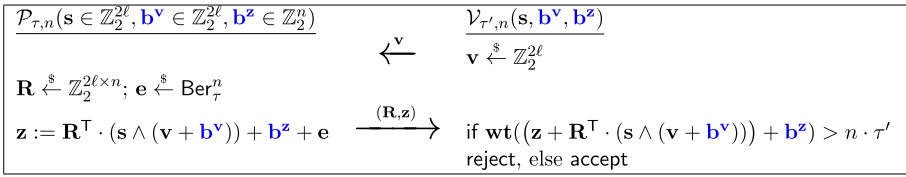


Fig. 5. By blinding the values \mathbf{v}, \mathbf{z} with secret random vectors $\mathbf{b}^v, \mathbf{b}^z$ we can avoid checking whether $\text{wt}(\mathbf{v}) = \ell$ and $\text{rank}(\mathbf{R}) = n$ as in the protocol from Fig. 4.

The setup phase is as before, except that \mathcal{B} apart from \mathbf{x}^* and \mathbf{v}^* also samples random \mathbf{b}^v and \mathbf{b}^z . In the first phase $\mathcal{B}^\mathcal{O}$ will answer \mathcal{A} 's queries as before, except that we use $\mathbf{v} + \mathbf{b}^v$ instead of \mathbf{v} and finally also add \mathbf{b}^v to the answer (just so when \mathcal{O} is an SLPN oracle we get the same distribution as in the protocol). Also the second phase is identical, except that we use the challenge $\mathbf{v}^* + \mathbf{b}^v$ (so the actual subset used to answer the challenge is $\mathbf{v}^* + \mathbf{b}^v + \mathbf{b}^v = \mathbf{v}^*$).

Let $\alpha_{\ell,d}^*$ denote the probability that a random string of length ℓ has Hamming weight less than d , then we have $\Pr[\mathcal{B}^{U_{2\ell+1}^{(\cdot)}} = 1] \leq \alpha'_{\tau',n} + Q \cdot \alpha_{\ell,d}^*$, this is basically as in Claim 3.2, except that $\alpha'_{\ell,d}$ is replaced with $\alpha_{\ell,d}^*$. As \mathbf{b}^v is uniformly random, the probability that $\text{wt}((\mathbf{v} + \mathbf{b}^v) \wedge \bar{\mathbf{v}}^*) < d$ is $\alpha_{\ell,d}^*$ (in which case the oracle will output \perp), using the union bound this will not happen in any of the Q queries except with probability at most $Q \cdot \alpha_{\ell,d}^*$. In this argument we also must use the fact that the answers that \mathcal{B} sends to \mathcal{A} are uniformly random, and thus do not leak any information about \mathbf{b}^v (except from what can be deduced from the fact that \mathcal{B} has not already output 1 and stopped). If we come to the second phase, the probability that \mathcal{A} comes up with a valid answer $(\mathbf{R}^*, \mathbf{z}^*)$ is at most $\alpha'_{\tau',n}$ as at this point \mathbf{b}^z , and thus also the string $\mathbf{w} = (\mathbf{z} + \mathbf{R}^\top \cdot (\mathbf{s} \wedge \mathbf{v}^*)) + \mathbf{b}^z$, is uniformly random given \mathcal{A} 's view, which implies that $\Pr[\text{wt}(\mathbf{w}) > n \cdot \tau'] \leq \alpha'_{\tau',n}$.

The proof that $\Pr[\mathcal{B}^{\Gamma_{\tau,2\ell,d}^*(\mathbf{x}^*)} = 1] \geq \epsilon'$ is almost analogous to the proof of Claim 3.3.

4. Message Authentication Codes

In this section, we construct two message authentication codes whose security can be reduced to the LPN assumption. Our first construction is based on the two-round authentication protocol from Sect. 3. We prove that if the LPN problem is ϵ -hard, then no adversary making Q queries can forge a MAC with probability more than $\Theta(\sqrt{\epsilon} \cdot Q)$. The construction is somewhat non-uniform as to achieve this security, one must set a parameter in the construction as a function of ϵ (cf. Remark 4.4). Our second construction has no such issues and achieves better security $\Theta(\epsilon \cdot Q)$. The efficiency of this construction is similar to that of the first construction, but a larger key is required.

4.1. First Construction

Recall the two-round authentication protocol from Sect. 3. In the protocol the verifier chooses a random challenge subset \mathbf{v} . To turn this interactive protocol into a MAC, we

will compute this \mathbf{v} from the message m to be authenticated as $\mathbf{v} = \mathbf{C}(h(m, \mathbf{b}))$, where h is a pairwise independent hash function, $\mathbf{b} \in \mathbb{Z}_2^v$ is some fresh randomness and \mathbf{C} is some encoding scheme. The code \mathbf{C} is fixed and public, while the function h is part of the secret key. The authentication tag ϕ is computed in the same manner as the prover's answer in the authentication protocol. That is, we sample a random matrix $\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}$ and compute a noisy inner product $\mathbf{z} := \mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e}$, where $\mathbf{e} \stackrel{\$}{\leftarrow} \text{Ber}_\tau^n$. We note that using $(\mathbf{R}, \mathbf{z}, \mathbf{b})$ as an authentication tag would not be secure, and we need to blind these values. This is done by applying a pairwise independent permutation (PIP) π —which is part of the secret key—to $(\mathbf{R}, \mathbf{z}, \mathbf{b}) \in \mathbb{Z}_2^{\ell \times n + n + v}$.

Remark 4.1. In the construction it is sufficient to just use *almost* pairwise independent functions and permutations. In fact, if the message space \mathcal{M} is a priori unbounded, we have to settle for almost pairwise independence (as the description of a pairwise independent h is linear in the length of the inputs it can process).

Construction. The message authentication code $\text{MAC}_1 = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ with associated message space \mathcal{M} is defined as follows:

- **Public Parameters.** MAC_1 has the following public parameters.⁷

ℓ, τ, τ', n	as in the authentication protocol from Sect. 3
$\mu \in \mathbb{N}$	output length of the hash function
$v \in \mathbb{N}$	length of the randomness
$\mathbf{C} : \mathbb{Z}_2^\mu \rightarrow \mathbb{Z}_2^{2\ell}$	encoding, where $\forall \mathbf{y} \neq \mathbf{y}' \in \mathbb{Z}_2^\mu$ we have $\text{wt}(\mathbf{C}(\mathbf{y})) = \ell$ and $\text{wt}(\mathbf{C}(\mathbf{y}) + \mathbf{C}(\mathbf{y}')) \geq 0.9\ell$.

- **Key Generation.** Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{s} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{2\ell}$, a pairwise independent hash function $h : \mathcal{M} \times \mathbb{Z}_2^v \rightarrow \mathbb{Z}_2^\mu$ and a pairwise independent permutation π over $\mathbb{Z}_2^{\ell \times n + n + v}$. It returns $K = (\mathbf{s}, h, \pi)$ as the secret key.
- **Tagging.** Given secret key $K = (\mathbf{s}, h, \pi)$ and message $m \in \mathcal{M}$, algorithm \mathcal{T} proceeds as follows.

1. $\mathbf{R} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell \times n}$, $\mathbf{b} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^v$, $\mathbf{e} \stackrel{\$}{\leftarrow} \text{Ber}_\tau^n$
2. $\mathbf{v} := \mathbf{C}(h(m, \mathbf{b})) \in \mathbb{Z}_2^{2\ell}$
3. Return $\phi := \pi(\mathbf{R}, \mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e}, \mathbf{b})$

- **Verification.** On input a secret key $K = (\mathbf{s}, h, \pi)$, message $m \in \mathcal{M}$ and tag ϕ , algorithm \mathcal{V} proceeds as follows.

1. Parse $\pi^{-1}(\phi)$ as $(\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^v)$. If $\text{rank}(\mathbf{R}) \neq n$, then return reject
2. $\mathbf{v} := \mathbf{C}(h(m, \mathbf{b}))$
3. If $\text{wt}(\mathbf{z} + \mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}}) > n \cdot \tau'$ return reject, otherwise return accept

⁷The code \mathbf{C} can be constructed as follows. We first sample a random matrix $\mathbf{C} \in \mathbb{Z}_2^{\mu \times \ell}$ and map $\mathbf{y} \in \mathbb{Z}_2^\mu$ to $\mathbf{C}(\mathbf{y}) = (\mathbf{c} \in \mathbb{Z}_2^\ell, \mathbf{c}' \in \mathbb{Z}_2^\ell)$ where $\mathbf{c} = \mathbf{C}^T \cdot \mathbf{y}$ and $\mathbf{c}' = \bar{\mathbf{c}}$. A random code \mathbf{C} has high distance with high probability and $\mathbf{C}(\mathbf{y}) = (\mathbf{c}, \mathbf{c}')$ has weight exactly ℓ .

Theorem 4.2. For $\mu = \nu \in \mathbb{N}$, a constant $\gamma > 0$ and $d := \ell / (2 + \gamma)$, if the $\text{SLPN}_{\tau, 2\ell, d}^*$ problem is (t, nQ, ε) -hard then MAC_1 is (t', Q, ε') -secure against uf-cma adversaries, where for constants $c_\gamma, c_\tau > 0$ that depend only on γ and τ , respectively, and for $Q = Q_{\text{tag}} + Q_{\text{vrfy}}$,

$$t' \approx t, \quad \varepsilon = \min \left\{ \varepsilon' / 2 - \frac{Q^2}{2^\mu}, \frac{\varepsilon'}{2^{\mu+1}} - Q_{\text{tag}} \cdot 2^{-c_\gamma \cdot \ell} - Q_{\text{vrfy}} \cdot 2^{-c_\tau \cdot n} \right\}.$$

MAC_1 has completeness error $2^{-c'_\tau \cdot n} + 2^{n-\ell} \in 2^{-\Theta(n)}$ where $c'_\tau > 0$ depends only on τ .

Corollary 4.3. Choosing μ s.t. $2^\mu = \frac{4Q^2}{\varepsilon'}$ in the above theorem, we get $\varepsilon = \min\{\varepsilon'/4, \varepsilon'^2/8Q^2 - Q \cdot 2^{-\Theta(n)}\}$. The second term is the minimum here, and solving for ε' gives

$$\varepsilon' \leq 3Q \cdot \sqrt{\varepsilon + Q \cdot 2^{-\Theta(n)}}. \tag{4.1}$$

Remark 4.4. (about μ) Note that to get security as claimed in the above corollary, we need to choose μ as a function of Q and ε such that $2^\mu \approx 4Q^2/\varepsilon'$ for ε' as in Eq. (4.1). Of course we can just fix Q (as an upper bound to the number of queries made by the adversary) and ε (as our guess on the actual hardness of $\text{SLPN}_{\tau, 2\ell, d}^*$). But a too conservative guess on μ (i.e., choosing μ too small) will result in a construction whose security is worse than what is claimed in the above corollary. A too generous guess on the other hand will make the security reduction meaningless, though we do not have any attacks on the MAC for large μ .

We now give some intuition for the proof of Theorem 4.2. Every query (m, ϕ) to \mathcal{V} and query m to \mathcal{T} defines a subset \mathbf{v} (as computed in the second step in the definitions of both \mathcal{V} and \mathcal{T}). We say that a forgery (m, ϕ) is “fresh” if the \mathbf{v} contained in (m, ϕ) is different from all \mathbf{v} ’s contained in all the previous \mathcal{V} and \mathcal{T} queries. The proof makes a case distinction and uses a different reduction for the two cases where the first forgery found by the adversary is more likely to be fresh, or more likely to be non-fresh. In both cases we consider a reduction $\mathcal{B}^\mathcal{O}$ which has access to either a uniform oracle $\mathcal{O} = U$ or a subset LPN oracle $\mathcal{O} = \Gamma^*$. Adversary $\mathcal{B}^\mathcal{O}$ uses an adversary \mathcal{A} who can find forgeries for the MAC to distinguish those cases and thus break the subset LPN assumption. In the first case, where the first forgery is likely to be **non-fresh**, we can show (using the fact that a pairwise independent permutation is used to blind the tag) that if $\mathcal{B}^\mathcal{O}$ ’s oracle is $\mathcal{O} = U$, even a computationally unbounded \mathcal{A} cannot come up with a message/tag pair (m, ϕ) that contains a non-fresh \mathbf{v} . Thus we can distinguish the cases $\mathcal{O} = U$ and $\mathcal{O} = \Gamma^*$ by just observing if \mathcal{A} ever makes a \mathcal{V} query (m, ϕ) that contains a non-fresh \mathbf{v} (even without being able to tell if (m, ϕ) is valid or not).

If the first forgery found by \mathcal{A} is more likely to be **fresh**, we can use a similar argument as in the proof of our authentication protocol in the last section. An additional difficulty here is that the reduction has to guess the fresh $\mathbf{v} \in \mathbb{Z}_2^{2\ell}$ (for which there are 2^μ possibilities) contained in the first forgery and cannot choose it itself as in the protocol. This is the reason why the reduction loses a factor 2^μ .

Proof of Theorem 4.2. The proof of completeness is essentially the same (and we get exactly the same quantitative bound) as the proof of completeness for the protocol in Fig. 4 as claimed in Theorem 3.1.

We now prove security. As in the theorem statement, we set $\mu = \nu$ (but for clarity we will keep the different letters μ for the range of h and ν for the length of the randomness). Let \mathcal{A} be an adversary running in time t' that breaks the uf-cma security of MAC_1 in the experiment $\text{Exp}_{\text{MAC}_1, \mathcal{A}, \lambda}^{\text{uf-cma}}$ with advantage ε' . Let Q_{tag} and Q_{vrfy} denote the number of queries that \mathcal{A} makes to the tag and verification oracles, respectively, such that $Q = Q_{\text{tag}} + Q_{\text{vrfy}}$. We assume that \mathcal{A} never makes the same verification query twice (since \mathcal{V} is deterministic, repeating queries gives no additional information to \mathcal{A}) and also that she never makes a verification query (m, ϕ) where ϕ was received as the output from the tag oracle on input m . Since the completeness error of MAC_1 is $2^{-\Theta(n)}$, this is basically without loss of generality (as the answer would almost certainly be `accept`). Every verification query (m, ϕ) and tag query m define a subset \mathbf{v} (as computed in step 2. in the definitions of both \mathcal{V} and \mathcal{T}).

By definition, in the uf-cma experiment, with probability ε' the adversary \mathcal{A} at some point makes a verification query (m, ϕ) where (i) ϕ was not received as output on a tag query m , and (ii) $\mathcal{V}(K, m, \phi) = \text{accept}$. We say that such a forgery (m, ϕ) is “fresh” if the \mathbf{v} defined by (m, ϕ) is different from all \mathbf{v} ’s defined by all the previous verification and tag queries. Let E_{fresh} denote the event that the first forgery found by \mathcal{A} is fresh. We will consider the two cases where $\Pr[E_{\text{fresh}}] > \varepsilon'/2$ and $\Pr[E_{\text{fresh}}] \leq \varepsilon'/2$ separately.

The case $\Pr[E_{\text{fresh}}] \leq \varepsilon'/2$. Given \mathcal{A} , we will construct an adversary $\mathcal{B}_1^{\mathcal{O}}$ who can distinguish $\mathcal{O} = \Gamma_{\tau, 2\ell, \ell}^*(\mathbf{s}, \cdot)$ from $\mathcal{O} = U_{2\ell+1}(\cdot)$ (as in Definition 2.8) with advantage

$$\varepsilon'/2 - \frac{Q^2}{2^\mu}. \quad (4.2)$$

$\mathcal{B}_1^{\mathcal{O}}$ samples π, h (but not \mathbf{s}) as defined by \mathcal{K} . Next, it invokes \mathcal{A} (who expects to attack MAC_1 with a key (\mathbf{s}, h, π)) answering its queries as follows:

- **Tag queries.** If \mathcal{A} makes a tag query m , then $\mathcal{B}_1^{\mathcal{O}}$ does the following:

1. Sample $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^\nu$ and compute $\mathbf{v} := \mathbf{C}(h(m, \mathbf{b}))$.
2. Query the oracle \mathcal{O} for n times on input \mathbf{v} : for $i = 1, \dots, n$ let $(\mathbf{R}[i], \mathbf{z}[i]) \xleftarrow{\$} \mathcal{O}(\mathbf{v})$.
3. Return $\phi := \pi(\mathbf{R}, \mathbf{z}, \mathbf{b})$ where $\mathbf{R} = [\mathbf{R}[1], \dots, \mathbf{R}[n]]$ and $\mathbf{z} = [\mathbf{z}[1], \dots, \mathbf{z}[n]]$ to \mathcal{A} .

- **Verification queries.** If \mathcal{A} makes a verification query (m, ϕ) , $\mathcal{B}_1^{\mathcal{O}}$ simply answers with `reject`.

If any tag or verification query contains a \mathbf{v} which has appeared in a previous query, $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 and 0 otherwise. (Note that $\mathcal{B}_1^{\mathcal{O}}$ can compute the value \mathbf{v} in a verification query as it knows π, h .)

Claim 4.5. *If $\mathcal{O} = \Gamma_{\tau, 2\ell, \ell}^*(\mathbf{s}, \cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $\geq \varepsilon'/2$.*

Proof of Claim. The answers to the tag queries of \mathcal{A} computed by $\mathcal{B}_1^{\mathcal{O}}$ have exactly the same distribution as in the uf-cma experiment (where the secret key is (s, h, π)). (Note that each tag query yields a vector \mathbf{v} with hamming weight exactly ℓ .) The answers to the verification queries (which are always **reject**) are correct as long as \mathcal{A} does not query a valid forgery. The probability that the first valid forgery found by \mathcal{A} is *not* fresh is $\varepsilon' - \Pr[E_{\text{fresh}}] \geq \varepsilon' - \varepsilon'/2 \geq \varepsilon'/2$, which is thus a lower bound on the probability that $\mathcal{B}_1^{\mathcal{O}}$ outputs 1. \square

Claim 4.6. *If $\mathcal{O} = U_{2\ell+1}(\cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $< Q^2/2^\mu$.*

Proof of Claim. The answers that \mathcal{A} obtains on a tag query m from $\mathcal{B}_1^{U_{2\ell+1}(\cdot)}$ (i.e., $\pi(\mathbf{R}, \mathbf{z}, \mathbf{b})$ where $\mathbf{R}, \mathbf{z}, \mathbf{b}$ are sampled uniformly) are uniformly random, and in particular independent of h or π . The answers to verification queries are always **reject** and thus contain no information about h, π either. Then, we have that $\mathbf{v}_i = \mathbf{v}_j$ (where $\mathbf{v}_i = \mathbf{C}(h(m_i, \mathbf{b}_i))$ is defined by the i th tag or verification query) iff $h(m_i, \mathbf{b}_i) = h(m_j, \mathbf{b}_j)$. We will upper bound the probability of such a collision, which then is also an upper bound on the probability that $\mathcal{B}_1^{\mathcal{O}}$ outputs 1. \mathcal{A} makes a total of Q queries. Assume that up to the $(i - 1)$ th query, all the \mathbf{v} 's were distinct.

If the i th query is a tag query m_i , a fresh \mathbf{b}_i is sampled which will collide with a previous \mathbf{b}_j ($j < i$) with probability at most $(i - 1)/2^\nu$. Assuming no such collision happened, the probability that $h(m_i, \mathbf{b}_i) = h(m_j, \mathbf{b}_j)$ for any $j < i$ can be upper bounded by $(i - 1)/2^\mu$ (here we use the fact that the answers that \mathcal{A} gets from $\mathcal{B}_1^{U_{2\ell+1}(\cdot)}$ are uniformly random, and thus \mathcal{A} has no information about the function h).

If the i th query is a verification query (m_i, ϕ_i) , then using the fact that π is a pairwise independent permutation (and \mathcal{A} has no information about it) we get that $\pi^{-1}(\phi_i) = (\mathbf{R}_i, \mathbf{z}_i, \mathbf{b}_i)$ contains a \mathbf{b}_i which will collide with a previous \mathbf{b}_j ($j < i$) where $\phi_j \neq \phi_i$ with probability at most $(i - 1)/2^\nu$. If we have no such collision, then also $(m_i, \mathbf{b}_i) \neq (m_j, \mathbf{b}_j)$ for all $j < i$ as for j where $\phi_i = \phi_j$ we must have that $m_i \neq m_j$.⁸ If $(m_i, \mathbf{b}_i) \neq (m_j, \mathbf{b}_j)$ for all $j < i$ we can again upper bound the probability that $h(m_i, \mathbf{b}_i) = h(m_j, \mathbf{b}_j)$ for some $j < i$ by $(i - 1)/2^\mu$.

We showed that for the i th query (no matter if it is a tag or verification query) the probability that $h(m_i, \mathbf{b}_i) = h(m_j, \mathbf{b}_j)$ for some $j \leq i$ is at most $\frac{(i-1)}{2^\nu} + \frac{(i-1)}{2^\mu}$. Taking the union bound over all $i, 1 \leq i \leq Q$, we can upper bound the probability that $h(m_i, \mathbf{b}_i) = h(m_j, \mathbf{b}_j)$ for any $i \neq j$ as $\sum_{i=1}^Q \left(\frac{(i-1)}{2^\nu} + \frac{(i-1)}{2^\mu} \right) \leq Q^2/2^\mu$ (recall that $\mu = \nu$) as claimed. \square

The case $\Pr[E_{\text{fresh}}] > \varepsilon'/2$. In this case, \mathcal{A} will make tag/verification queries, where with probability $> \varepsilon'/2$, at some point she will make an accepting verification query (m, ϕ) that defines a fresh \mathbf{v} . We now construct an adversary $\mathcal{B}_2^{\mathcal{O}}$ that uses \mathcal{A} as a black-box, and can distinguish $\mathcal{O} = \Gamma_{\tau, 2\ell, d}^*(s, \cdot)$ from $\mathcal{O} = U_{2\ell+1}(\cdot)$ (as in Definition 2.8)

⁸Recall that we assume that \mathcal{A} does not repeat queries and does not ask verification queries (m, ϕ) if ϕ was the output of a tag query m .

with advantage

$$\frac{\varepsilon'}{2^{\mu+1}} - Q_{\text{tag}} \cdot \alpha'_{\ell,d} - Q_{\text{vrfy}} \cdot \alpha''_{\tau',n}. \quad (4.3)$$

The construction of $\mathcal{B}_2^{\mathcal{O}}$ is very similar to the adversary \mathcal{B} that we constructed in the proof of Theorem 3.1 (where we proved that the authentication protocol in Fig. 4 is secure against active attacks). The queries to the prover in the first phase of an active attack directly correspond to tag queries. However, we now have to additionally answer verification queries (we will always answer **reject**). Furthermore, we cannot choose the challenge \mathbf{v}^* . Instead, we will simply hope that (in the case where $\mathcal{O} = \Gamma_{\tau,2\ell,d}^*(\mathbf{s}, \cdot)$) the \mathbf{v} contained in the first valid verification query (i.e., forgery) that \mathcal{A} makes is fresh (which by assumption happens with probability $\varepsilon'/2$). Moreover, we will hope that it is the unique \mathbf{v}^* (out of 2^μ possible ones) for which $\mathcal{B}_2^{\mathcal{O}}$ can verify this. This gives us a distinguishing advantage of nearly $(\varepsilon'/2)/2^\mu = \varepsilon'/2^{\mu+1}$ as stated in Eq. (4.3). We do lose an additional additive term $Q_{\text{tag}} \cdot \alpha'_{\ell,d}$ as there is an exponentially small probability that the transformation of subspace LPN samples to tag queries will fail, and moreover an exponentially small term $Q_{\text{vrfy}} \cdot \alpha''_{\tau',n}$ which accounts for the probability that \mathcal{A} correctly guesses an accepting tag even in the case where $\mathcal{O} = U_{2\ell+1}(\cdot)$.

$\mathcal{B}_2^{\mathcal{O}}$ samples π, h (but not \mathbf{s}) as defined by \mathcal{K} , and $\mathbf{y}^* \xleftarrow{\$} \mathbb{Z}_2^\mu$, $\mathbf{s}^* \xleftarrow{\$} \mathbb{Z}_2^{2\ell}$. Let $\mathbf{v}^* := \mathbf{C}(\mathbf{y}^*)$. Next, $\mathcal{B}_2^{\mathcal{O}}$ invokes \mathcal{A} and answers its queries as follows (the intuition for the sampling below is given in the proof of Claim 4.8).

- **Tag queries.** The answer ϕ to a tag query $m \in \mathcal{M}$ is computed by $\mathcal{B}_2^{\mathcal{O}}$ as follows:
 1. Sample $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^v$ and compute $\mathbf{v} := \mathbf{C}(h(m, \mathbf{b}))$.
Let $\mathbf{u} := \mathbf{v} \wedge \bar{\mathbf{v}}^*$ and $\mathbf{u}^* := \mathbf{v} \wedge \mathbf{v}^*$.
 2. For $i = 1, \dots, n$, let $(\mathbf{R}'[i], \mathbf{z}'[i]) \xleftarrow{\$} \mathcal{O}(\mathbf{u})$, $\mathbf{R}''[i] \xleftarrow{\$} \mathbb{Z}_2^{2\ell}$ and $\mathbf{z}''[i] := (\mathbf{R}''[i], \mathbf{s}^* \wedge \mathbf{u}^*)$. Define $\mathbf{R} = [\mathbf{R}[1], \mathbf{R}[2], \dots, \mathbf{R}[n]]$ and $\mathbf{z} = [\mathbf{z}[1], \dots, \mathbf{z}[n]]$ where $\mathbf{R}[i] := (\mathbf{R}'[i] \wedge \mathbf{u} + \mathbf{R}''[i] \wedge \mathbf{u}^*)_{\downarrow \mathbf{v}}$ and $\mathbf{z}[i] := \mathbf{z}'[i] + \mathbf{z}''[i]$.
 3. Return $\phi := \pi(\mathbf{R}, \mathbf{z}, \mathbf{b})$ to \mathcal{A} .
- **Verification queries.** If \mathcal{A} makes a verification query (ϕ, m) , then $\mathcal{B}_2^{\mathcal{O}}$ always answers **reject**, but also makes the following check:
 1. Parse $\mathbf{y} := \pi^{-1}(\phi)$ as $[\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^v]$ and compute $\mathbf{v} := \mathbf{C}(h(m, \mathbf{b}))$.
 2. If $\mathbf{v} \neq \mathbf{v}^*$, processing this query is over, otherwise go to the next step.
 3. If $\text{rank}(\mathbf{R}) = n$ and $\text{wt}(\mathbf{R}^T \cdot \mathbf{s}_{\downarrow \mathbf{v}^*}^* + \mathbf{z}) \leq n \cdot \tau'$ (i.e., we have a forgery) output 1 and stop.

If \mathcal{A} has finished its queries, $\mathcal{B}_2^{\mathcal{O}}$ stops with output 0.

Claim 4.7. *If $\mathcal{O} = U_{2\ell+1}(\cdot)$, then $\mathcal{B}_2^{\mathcal{O}}$ outputs 1 with probability $\leq Q_{\text{vrfy}} \cdot \alpha''_{\tau',n} + Q_{\text{tag}} \cdot \alpha'_{\ell,d}$.*

Proof of Claim. The proof of this claim is almost identical to the proof of Claim 3.2, except that here we have an additional factor Q_{vrfy} as we have to take the union bound over all Q_{vrfy} queries, whereas in Claim 3.2 the adversary was (by definition of an active attack) only allowed one guess. \square

Claim 4.8. *If $\mathcal{O} = \Gamma_{\tau, 2\ell, d}^*(\mathbf{s}, \cdot)$, then $\mathcal{B}_2^{\mathcal{O}}$ outputs 1 with probability $\geq \frac{\varepsilon'}{2^{\mu+1}}$.*

Proof. The proof of this claim is similar to the proof of Claim 3.3. That is, one shows that \mathcal{B}_2 with oracle access to $\Gamma_{\tau, 2\ell, 0}^*(\mathbf{s}, \cdot)$ (i.e., the subset LPN oracle accepts subsets of arbitrarily small size) perfectly simulates the experiment and thus receive a fresh forgery from \mathcal{A} with probability at least $\varepsilon'/2$. Unlike in Claim 3.3, here \mathcal{B}_2 will only be able to recognize a fresh forgery if it is using a \mathbf{v} that is equivalent to \mathbf{v}^* , this accounts for the extra $2^{-\mu}$ factor. □

Summing up, using \mathcal{A} we can break the subset LPN assumption with advantage which is given either by Eq. (4.2) or Eq. (4.3), i.e.,

$$\varepsilon = \min \left\{ \varepsilon'/2 - \frac{2Q^2}{2^\mu}, \frac{\varepsilon'}{2^{\mu+1}} - Q_{\text{tag}} \cdot \alpha'_{\ell, d} - Q_{\text{vrfy}} \cdot \alpha''_{\tau', n} \right\}.$$

□

4.2. Second Construction

We now give the construction of another MAC based on the hardness of the LPN problem. The main difference from MAC_1 from the last subsection is the way we generate the values $\mathbf{s}(\mathbf{v})$. In the new construction we define $\mathbf{s}(\mathbf{v}) = \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$, where $\mathbf{S} \in \mathbb{Z}_2^{\ell \times \mu}$ and $\mathbf{s}_0 \in \mathbb{Z}_2^\ell$ are both part of the secret key. Moreover, in the computation of a tag, the output is masked via another vector $\mathbf{s}'_0 \in \mathbb{Z}_2^n$ that is also included in the secret key. The construction borrows ideas from [4] that we needed to adapt to the case of LPN.

Construction. The message authentication code $\text{MAC}_2 = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ with associated message space \mathcal{M} is defined as follows:

- **Public Parameters.** MAC_2 has the following public parameters.

ℓ, τ, τ', n as in the authentication protocol from Sect. 3

$\mu \in \mathbb{N}$ output length of the hash function

$\nu \in \mathbb{N}$ length of the randomness

- **Key Generation.** Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{S} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times \mu}$, $\mathbf{s}_0 \xleftarrow{\$} \mathbb{Z}_2^\ell$, $\mathbf{s}'_0 \xleftarrow{\$} \mathbb{Z}_2^n$ and chooses an almost pairwise independent hash function $h : \mathcal{M} \times \mathbb{Z}_2^\nu \rightarrow \mathbb{Z}_2^\mu \setminus \{0\}$, as well as a pairwise independent permutation π over $\mathbb{Z}_2^{\ell \times n + n + \nu}$ (cf. Remark 4.1). It returns $K = (\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0, h, \pi)$ as the secret key.
- **Tagging.** Given secret key $K = (\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0, h, \pi)$ and message $m \in \mathcal{M}$, algorithm \mathcal{T} proceeds as follows.

1. $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^\nu$, $\mathbf{e} \xleftarrow{\$} \text{Ber}_\tau^n$
2. $\mathbf{v} := h(m, \mathbf{b})$
3. $\mathbf{s}(\mathbf{v}) := \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$
4. Return $\phi := \pi(\mathbf{R}, \mathbf{s}'_0 + \mathbf{R}^\top \cdot \mathbf{s}(\mathbf{v}) + \mathbf{e}, \mathbf{b})$

- **Verification.** On input a secret key $K = (\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0, h, \pi)$, message $m \in \mathcal{M}$ and tag ϕ , algorithm \mathcal{V} proceeds as follows.
 1. Parse $\pi^{-1}(\phi)$ as $(\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^v)$. If $\text{rank}(\mathbf{R}) \neq n$, then return reject
 2. $\mathbf{v} := h(m, \mathbf{b})$
 3. $\mathbf{s}(\mathbf{v}) := \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$
 4. If $\text{wt}(\mathbf{z} + \mathbf{s}'_0 + \mathbf{R}^T \cdot \mathbf{s}(\mathbf{v})) > n \cdot \tau'$ return reject, otherwise return accept

Theorem 4.9. For $\mu = v \in \mathbb{N}$, if the $\text{SLPN}_{\tau, \ell, \ell}$ problem is (t, nQ, ε) -hard (by Proposition 2.3 this problem is as hard as $\text{LPN}_{\tau, \ell}$), then MAC_2 is (t', Q, ε') -secure against uf-cma adversaries, where for constants $c_\tau, c'_\tau > 0$ that depend only on τ , and for $Q = Q_{\text{tag}} + Q_{\text{vrfy}}$,

$$t' \approx t \quad \varepsilon = \min \left\{ \varepsilon' / 2 - \frac{Q^2}{2^\mu}, \frac{\varepsilon'}{8\mu Q_{\text{vrfy}}} - Q_{\text{vrfy}} \cdot (2^{-c'_\tau \cdot n} + 2^{-c_\tau \cdot n}) \right\}.$$

MAC_2 has completeness error $2^{-c'_\tau \cdot n} + 2^{n-\ell}$.

We now give intuition for the proof of Theorem 4.9. Similar to the proof of Theorem 4.2, we distinguish fresh and non-fresh forgeries. Here the new and interesting case is when the first forgery found by the adversary is fresh. In the analysis we move to a mental experiment where tags computed by the tag oracle are uniform and independent from the secret key. The technical heart of the proof is to show that such a modification defines an indistinguishable distribution, assuming that the standard LPN assumption holds. More in detail, consider the two experiments defined in Fig. 6. In the “real experiment,” the answers from the $\text{EVAL}(\mathbf{v})$ oracle have the same distribution as the values (\mathbf{R}, \mathbf{z}) from a tag on message m such that $h(m, \mathbf{b}) = \mathbf{v}$; in the “random experiment,” the answers from the $\text{EVAL}(\mathbf{v})$ oracle are uniform. Oracle $\text{CHAL}(\mathbf{R}^*, \mathbf{v}^*)$, which can be queried at most once, essentially corresponds to the output of a verification query on a fresh forgery (m^*, ϕ^*) , such that $h(m^*, \mathbf{b}^*) = \mathbf{v}^*$. The lemma below states that it is hard to distinguish the two cases. Its proof uses a hybrid technique from [4, 13].

Lemma 4.10. Let $\ell, \mu, n, \tau \in \mathbb{N}$. Assume that the $\text{LPN}_{\tau, \ell}$ problem is (t, nQ, ε) -hard. Then, for all adversaries \mathcal{B} running in time $t' \approx t$, and asking Q queries to the $\text{EVAL}(\cdot)$ oracle, we have that

$$\left| \Pr \left[\text{Exp}_{\ell, \mu, n, \tau}^{\text{real}}(\mathcal{B}) = 1 \right] - \Pr \left[\text{Exp}_{\ell, \mu, n, \tau}^{\text{rand}}(\mathcal{B}) = 1 \right] \right| \leq 4\mu\varepsilon.$$

Proof. We start by making a syntactical change in the real experiment. Let $\mathbf{S} = (\mathbf{S}[j])_{j \in [\mu]}$, with $\mathbf{S}[j] \in \mathbb{Z}_2^\ell$. One can show that there exist vectors $\mathbf{s}_{j,k} \in \mathbb{Z}_2^\ell$ for $j \in [\mu]$ and $k \in \{0, 1\}$ such that

$$\mathbf{s}(\mathbf{v}) = \mathbf{S} \cdot \mathbf{v} + \mathbf{s}_0 = \sum_{j=1}^{\mu} \mathbf{s}_{j, \mathbf{v}[j]}.$$

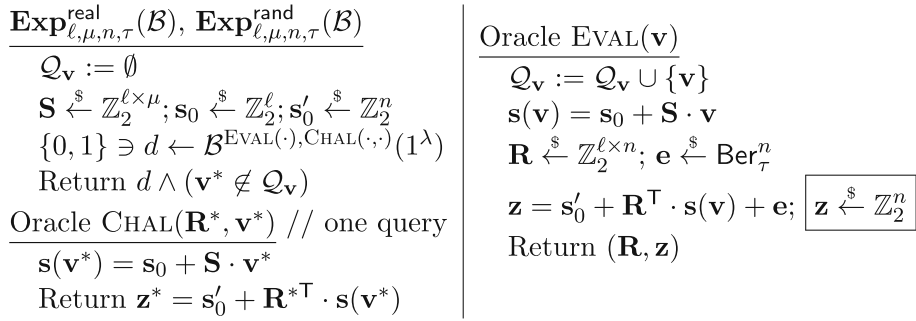


Fig. 6. Experiments $\text{Exp}_{\ell,\mu,n,\tau}^{\text{real}}(\mathcal{B})$ and $\text{Exp}_{\ell,\mu,n,\tau}^{\text{rand}}(\mathcal{B})$. The boxed statement redefining \mathbf{z} is only executed in $\text{Exp}_{\ell,\mu,n,\tau}^{\text{rand}}$.

This is obtained by letting $\mathbf{s}_0 = \sum_{j=1}^\mu \mathbf{s}_{j,0}$, and $\mathbf{S}[j] = \mathbf{s}_{j,1} - \mathbf{s}_{j,0}$. Furthermore, choosing the $\mathbf{s}_{j,k}$ uniformly at random yields uniformly distributed values \mathbf{S}, \mathbf{s}_0 .

Let \mathbf{G}_0 be identical to the “real experiment” $\text{Exp}_{\ell,\mu,n,\tau}^{\text{real}}(\mathcal{B})$, with the difference that at the beginning of the experiment we sample the vectors $\mathbf{s}_{i,j}$ (as defined above) uniformly at random, instead of sampling $(\mathbf{S}, \mathbf{s}_0)$, and later those vectors are used to define $\mathbf{s}(\mathbf{v})$. Clearly, \mathbf{G}_0 is identically distributed to the “real experiment.”

We prove the lemma by considering a sequence of intermediate games, starting with game \mathbf{G}_0 . The games are shown in Fig. 7. Note that in Game $\mathbf{G}_{1,0}$ the value \mathbf{s}'_0 is computed as $\text{RF}_0(\perp)$, where \perp is the empty string, that always outputs the same vector $\mathbf{s}'_0 \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\ell$. Therefore, we have

Claim 4.11. $\Pr[\mathbf{G}_0 = 1] = \Pr[\mathbf{G}_{1,0} = 1]$.

The next claim shows that any two adjacent hybrid games are indistinguishable, if the LPN assumption holds.

Claim 4.12. For all $i \in \{1, \dots, \mu - 1\}$, there exists a distinguisher \mathcal{D} , with running time similar to that of \mathcal{B} , such that

$$\left| \Pr \left[\mathcal{D}^{\Lambda_{\tau,\ell}(\mathbf{s})} = 1; \mathbf{s} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\ell \right] - \Pr \left[\mathcal{D}^{U_{\ell+1}} = 1 \right] \right| \geq \frac{1}{2} \left| \Pr [\mathbf{G}_{1,i+1} = 1] - \Pr [\mathbf{G}_{1,i} = 1] \right|.$$

Proof. Fix some particular $i \in [\mu - 1]$, and let Q be the total number of queries that \mathcal{B} asks to the $\text{EVAL}(\cdot)$ oracle. Distinguisher $\mathcal{D}^{\mathcal{O}}$ will ask nQ queries to its oracle $\mathcal{O}(\cdot)$, where \mathcal{O} is either equal to $\Lambda_{\tau,\ell}(\mathbf{s})$ or to $U_{\ell+1}$. Looking ahead, distinguisher \mathcal{D} will start by sampling a bit $b \stackrel{\$}{\leftarrow} \{0, 1\}$ as its guess for $\mathbf{v}^*[i + 1]$ and defining the random function $\text{RF}_{i+1}(\cdot)$ recursively as

$$\text{RF}_{i+1}(\mathbf{v}[1 \dots i + 1]) = \begin{cases} \text{RF}_i(\mathbf{v}[1 \dots i]) & \text{if } \mathbf{v}[i + 1] = b \\ \text{RF}_i(\mathbf{v}[1 \dots i]) + \text{RF}'_i(\mathbf{v}[1 \dots i]) & \text{otherwise} \end{cases} \quad (4.4)$$

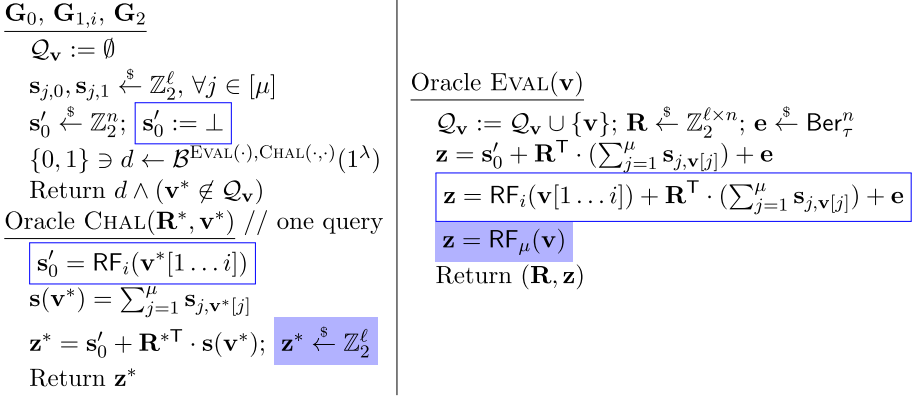


Fig. 7. Hybrid experiments \mathbf{G}_0 , \mathbf{G}_0 , $\mathbf{G}_{1,i}$ and \mathbf{G}_2 in the proof of Lemma 4.10. Here $\text{RF}_i : \{0, 1\}^i \rightarrow \mathbb{Z}_2^\ell$ is a random function, and $\mathbf{v}[1 \dots i] \in \mathbb{Z}_2^i$, for $i \in [\mu]$, is the i th prefix of vector $\mathbf{v} \in \mathbb{Z}_2^\ell$.

where $\text{RF}'_i : \{0, 1\}^i \rightarrow \mathbb{Z}_2^\ell$ is another random function (to be determined). One can verify that, in case $\text{RF}_i(\cdot)$ is a random function, so is $\text{RF}_{i+1}(\cdot)$. A formal description of \mathcal{D} follows:

1. At setup \mathcal{D} does the following:
 - Sample $b \stackrel{\$}{\leftarrow} \{0, 1\}$ and set $\mathcal{Q}_v := \emptyset$.
 - Choose all the vectors $\mathbf{s}_{j,k} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\ell$ at random (for all $j \in [\mu]$ and $k \in \{0, 1\}$), except for $\mathbf{s}_{i+1, 1-b}$ which is implicitly set to be the vector \mathbf{s} from the LPN oracle.
 - Query the $\mathcal{O}(\mathbf{s})$ oracle for nQ times, obtaining answers $(\mathbf{R}_j, \mathbf{z}'_j)_{j \in [Q]}$; let \mathcal{L}_i be an initially empty list.
2. Upon input a query \mathbf{v} to oracle $\text{EVAL}(\cdot)$, distinguisher \mathcal{D} does the following:
 - Update $\mathcal{Q}_v := \mathcal{Q}_v \cup \{\mathbf{v}\}$.
 - If $\mathbf{v}[i+1] = b$, sample $\mathbf{R} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell \times n}$ and $\mathbf{e} \stackrel{\$}{\leftarrow} \text{Ber}_\tau^n$, compute $\mathbf{z} = \text{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^{\text{T}} \cdot (\sum_{j=1}^{\mu} \mathbf{s}_{j, \mathbf{v}[j]}) + \mathbf{e}$, and return (\mathbf{R}, \mathbf{z}) .
 - Else, in case $\mathbf{v}[i+1] = 1-b$, check whether there exists an entry of the form $(\mathbf{v}[1, \dots, i], (\mathbf{R}_j, \mathbf{z}'_j))$ in the list \mathcal{L}_i , for some $j \in [Q]$. If this is the case, let $(\mathbf{R}, \mathbf{z}') := (\mathbf{R}_j, \mathbf{z}'_j)$, otherwise let $(\mathbf{R}, \mathbf{z}') := (\mathbf{R}_j, \mathbf{z}'_j)$ for the next (in lexicographic order) not already used LPN sample, and add $(\mathbf{v}[1, \dots, i], (\mathbf{R}_j, \mathbf{z}'_j))$ to the list \mathcal{L}_i . Define

$$\mathbf{z} = \text{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^{\text{T}} \cdot \sum_{\substack{j=1 \\ j \neq i+1}}^{\mu} \mathbf{s}_{j, \mathbf{v}[j]} + \mathbf{z}'$$

and return (\mathbf{R}, \mathbf{z}) .

3. Upon input query $(\mathbf{R}^*, \mathbf{v}^*)$ to oracle $\text{CHAL}(\cdot)$, distinguisher \mathcal{D} does the following:

- In case $\mathbf{v}^*[i + 1] \neq b$, abort.
- Else, define $\mathbf{s}(\mathbf{v}^*) = \sum_{j=1}^{\mu} \mathbf{s}_{j, \mathbf{v}^*[j]}$.
- Return $\mathbf{z}^* = \mathbf{R}^{*\top} \cdot \mathbf{s}(\mathbf{v}^*) + \text{RF}_i(\mathbf{v}^*[1 \dots i])$.

4. Upon input the decision bit d from \mathcal{B} , distinguisher \mathcal{D} returns $d \wedge (\mathbf{v}^* \notin \mathcal{Q}_{\mathbf{v}})$.

Suppose that \mathcal{D} correctly guessed $\mathbf{v}^*[i + 1]$, which happens with probability $1/2$. Note that in this case \mathcal{D} simulates perfectly the answer of the $\text{CHAL}(\cdot)$ oracle (as it knows $\mathbf{s}_{i+1, b}$). It remains to analyze the distribution of oracle $\text{EVAL}(\cdot)$. Below we argue that, depending on the value of $\mathbf{v}[i + 1]$, the distribution will either equal that of \mathbf{G}_i or \mathbf{G}_{i+1} , with random functions RF_{i+1} as defined in Eq. (4.4). In case $\mathbf{v}[i + 1] = b$, then the distribution is equal to that of both \mathbf{G}_i and \mathbf{G}_{i+1} (which is the same, as in this case $\text{RF}_{i+1}(\mathbf{v}[1 \dots i + 1]) = \text{RF}_i(\mathbf{v}[1 \dots i])$). In case $\mathbf{v}[i + 1] = 1 - b$, we consider two cases depending on whether the oracle $\mathcal{O}(\mathbf{s})$ outputs LPN samples or uniform samples. In the first case, we have $\mathbf{z}' = \mathbf{R}^{\top} \cdot \mathbf{s}_{i+1, 1-b} + \mathbf{e}$ and thus the answer

$$\begin{aligned} \mathbf{z} &= \text{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^{\top} \cdot \sum_{\substack{j=1 \\ j \neq i+1}}^{\mu} \mathbf{s}_{j, \mathbf{v}[j]} + \mathbf{R}^{\top} \cdot \mathbf{s}_{i+1, 1-b} + \mathbf{e} \\ &= \text{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^{\top} \cdot \sum_{j=1}^{\mu} \mathbf{s}_{j, \mathbf{v}[j]} + \mathbf{e}, \end{aligned}$$

is distributed like in game \mathbf{G}_i . In the second case, we have $\mathbf{z}' = \mathbf{R}^{\top} \cdot \mathbf{s}_{i+1, 1-b} + \mathbf{e} + \mathbf{u}$ (for a uniform $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_2^n$). Thus, the answer \mathbf{z} computed by \mathcal{D} is distributed like in \mathbf{G}_{i+1} with random function $\text{RF}'_i(\mathbf{v}[1 \dots i]) = \mathbf{u}$ and RF_{i+1} defined as in Eq. (4.4). Note that RF'_i is well defined, i.e., the value $\text{RF}'_i(\mathbf{v}[1, \dots, i])$ does not get overwritten in case the $\text{EVAL}(\cdot)$ oracle is queried on two different \mathbf{v}, \mathbf{v}' that are equal in the first i positions (this is because the reduction keeps track of which LPN sample is associated to each of the vectors $\mathbf{v}[1 \dots i]$, using the list \mathcal{L}_i). The claim follows. \square

Claim 4.13. $\Pr[\mathbf{G}_{1, \mu} = 1] = \Pr[\mathbf{G}_2 = 1]$.

Proof. The claim follows from the fact that in $\mathbf{G}_{1, \mu}$ all outputs computed via $\text{EVAL}(\cdot)$ are masked by $\text{RF}_{\mu}(\mathbf{v})$ and thus are independent of $\mathbf{s}_{j, k}$. Hence, the output of $\text{CHAL}(\cdot)$ is uniform.

Finally, we make all steps in reverse order to re-obtain the initial distribution in the $\text{EVAL}(\cdot)$ oracle. The proof of the following claim is analogous to the one of Claim 4.12 and is therefore omitted.

Claim 4.14. For all $i \in \{1, \dots, \mu - 1\}$, there exists a distinguisher \mathcal{D} , with running time similar to that of \mathcal{B} , such that

$$\begin{aligned} &\left| \Pr \left[\mathcal{D}^{\Lambda_{\tau, \ell}(\mathbf{s})} = 1; \mathbf{s} \xleftarrow{\$} \mathbb{Z}_2^{\ell} \right] - \Pr \left[\mathcal{D}^{U_{\ell+1}} = 1 \right] \right| \\ &\geq \frac{1}{2} \left| \Pr[\mathbf{G}_2 = 1] - \Pr \left[\text{Exp}_{\ell, \mu, n, \tau}^{\text{rand}}(\mathcal{B}) = 1 \right] \right|. \end{aligned}$$

The statement of Lemma 4.10 now follows by putting together Claims 4.11—4.14. \square

We now turn to the proof of Theorem 4.9.

Proof of Theorem 4.9. The proof of the completeness error is similar to the schemes before and is therefore omitted. As for security, let \mathcal{A} be an adversary that successfully forges in the uf-cma experiment with probability ε' . Assume that \mathcal{A} asks a total of $Q = Q_{\text{tag}} + Q_{\text{vrfy}}$ queries, where Q_{tag} (resp., Q_{vrfy}) stands for the total number of queries asked to the tag (resp., verification) oracle.

We make the same conventions and the definition of freshness as in the proof of Theorem 4.2 and consider the two cases $\Pr[E_{\text{fresh}}] \leq \varepsilon'/2$ and $\Pr[E_{\text{fresh}}] > \varepsilon'/2$ separately.

The case $\Pr[E_{\text{fresh}}] \leq \varepsilon'/2$. We now give the description of $\mathcal{B}_1^{\mathcal{O}}$ attacking the $\text{SLPN}_{\tau,\ell,\ell}$ problem, i.e., $\mathcal{B}_1^{\mathcal{O}}$ can distinguish $\mathcal{O} = \Gamma_{\tau,\ell,\ell}(\mathbf{s}, \cdot, \cdot)$ from $\mathcal{O} = U_{\ell+1}(\cdot, \cdot)$ with advantage

$$\varepsilon'/2 - \frac{Q^2}{2^\mu}. \quad (4.5)$$

Adversary $\mathcal{B}_1^{\mathcal{O}}$ samples $\pi, h, \mathbf{S}, \mathbf{s}'_0$ (but not \mathbf{s}_0) as defined by \mathcal{K} and $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times \mu}$. Next, it implicitly defines \mathbf{s}_0 as $\mathbf{s}_0 = \mathbf{s}$, where \mathbf{s} is only implicitly defined through $\Gamma_{\tau,\ell,\ell}(\mathbf{s}, \cdot, \cdot)$. It is easy to see that with this setup of $K = (\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0, h, \pi)$ we have that, for each $\mathbf{v} \in \mathbb{Z}_2^\mu \setminus \{0\}$,

$$\mathbf{s}(\mathbf{v}) = \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v} = \mathbf{s} + \mathbf{S} \cdot \mathbf{v}. \quad (4.6)$$

Note that $\mathbf{S} \cdot \mathbf{v}$ is known to \mathcal{B}_1 . Adversary $\mathcal{B}_1^{\mathcal{O}}$ cannot evaluate $\mathbf{s}(\mathbf{v})$ but looking ahead, it will use its oracle \mathcal{O} to answer \mathcal{A} 's queries as follows.

- **Tag queries.** If \mathcal{A} makes a tag query for message $m \in \mathcal{M}$, then $\mathcal{B}_1^{\mathcal{O}}$ does the following:
 1. Samples $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^\nu$ and compute $\mathbf{v} := h(m, \mathbf{b})$.
 2. Query the oracle \mathcal{O} on $(\mathbf{I}, \mathbf{S} \cdot \mathbf{v})$ for n times to obtain $(\mathbf{R}, \mathbf{z}')$: for $i = 1, \dots, n$ let $(\mathbf{R}[i], \mathbf{z}'[i]) \xleftarrow{\$} \mathcal{O}(\mathbf{I}, \mathbf{S} \cdot \mathbf{v})$. Here \mathbf{I} is the identity matrix.
 3. Return $\phi := \pi(\mathbf{R}, \mathbf{s}'_0 + \mathbf{z}', \mathbf{b})$.
- **Verification queries.** If \mathcal{A} makes a verification query (m, ϕ) , $\mathcal{B}_1^{\mathcal{O}}$ simply answers with reject.

Finally, if any tag or verification query contains a \mathbf{v} which has appeared in a previous query, $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 and stops. Otherwise, it outputs 0. Note that if $\mathcal{O} = \Gamma_{\tau,\ell,\ell}(\mathbf{s}, \cdot, \cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ perfectly simulates the $\mathcal{T}(K, \cdot)$ algorithm, as $\mathbf{z}'[i] = \mathbf{R}[i]^\top (\mathbf{I} \cdot \mathbf{s} + \mathbf{S} \cdot \mathbf{v}) + \mathbf{e}[i] = \mathbf{R}[i]^\top \cdot \mathbf{s}(\mathbf{v}) + \mathbf{e}[i]$.

The following two claims are the analogs of Claim 4.5 and 4.6, respectively. Their proofs are essentially the same and are therefore omitted.

Claim 4.15. *If $\mathcal{O} = \Gamma_{\tau,\ell}(\mathbf{s}, \cdot, \cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $\geq \varepsilon'/2$.*

Claim 4.16. *If $\mathcal{O} = U_{\ell+1}(\cdot, \cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $< \frac{Q^2}{2^\mu}$.*

The case $\Pr[E_{\text{fresh}}] > \varepsilon'/2$. We will use games, denoting by \mathbf{G}_i the output of the i th game. Game \mathbf{G}_0 runs the uf-cma security experiment $\text{Exp}_{\text{MAC}_2, \mathcal{A}, \lambda}^{\text{uf-cma}}$ and defines the output as 1 iff event E_{fresh} holds. By definition we have $\Pr[\mathbf{G}_0 = 1] = \Pr[E_{\text{fresh}}] > \varepsilon'/2$. Throughout the rest of the proof, if in the game \mathcal{A} finds a forgery, and the first forgery is fresh, we will denote with \cdot^* the values associated with this first forgery. In particular, \mathbf{v}^* is the \mathbf{v} -value computed to evaluate the verification query on (m^*, ϕ^*) . Note that, by definition, \mathbf{v}^* is fresh, i.e., it is different from all the \mathbf{v} -values from previous tag and verification queries. Without loss of generality, we assume that after \mathbf{G}_0 processes a verification query with respect to \mathbf{v}^* , as defined above, the random variable corresponding to the outcome of the game is defined and the experiment stops without processing any further query. For simplicity, we also assume that the answer to all verification queries (m, ϕ) before the forgery is found is `reject`, unless the pair (m, ϕ) was returned by the tag oracle (in which case we assume the answer is always `accept`); this results in a loss of at most $Q_{\text{vrfy}} \cdot \alpha$ in the final bound, where $\alpha := 2^{n-\ell} + 2^{-c'_t \cdot n}$ is the completeness error of MAC_2 .

Consider games $\mathbf{G}_1, \dots, \mathbf{G}_{Q_{\text{vrfy}}}$ where game \mathbf{G}_j is identical to \mathbf{G}_0 , but allows the adversary \mathcal{A} to ask only j verification queries. Define E_{fresh}^j to be the event that in \mathbf{G}_0 the j th verification query is the one where the first fresh forgery is found; this means that all previous verification queries are either rejected, or relative to a pair (m, ϕ) previously returned by the tag oracle. Since, for $j \in \{1, \dots, Q_{\text{vrfy}}\}$, all the events E_{fresh}^j are disjoint, and additionally $\Pr[\mathbf{G}_j = 1] \geq \Pr[E_{\text{fresh}}^j]$, we have:

$$\begin{aligned} \varepsilon'/2 < \Pr[E_{\text{fresh}}] = \Pr[\mathbf{G}_0 = 1] &= \Pr\left[\bigcup_{j=1}^{Q_{\text{vrfy}}} E_{\text{fresh}}^j\right] = \sum_{j=1}^{Q_{\text{vrfy}}} \Pr[E_{\text{fresh}}^j] \\ &\leq \sum_{j=1}^{Q_{\text{vrfy}}} \Pr[\mathbf{G}_j = 1]. \end{aligned} \quad (4.7)$$

In the remainder of the proof, we will upper bound $\Pr[\mathbf{G}_j = 1]$, for all $j \in \{1, \dots, Q_{\text{vrfy}}\}$. As a first step, consider a modified version \mathbf{G}'_j of Game \mathbf{G}_j where the tag oracle internally uses uniform $\mathbf{z} \in \mathbb{Z}_2^n$, instead of $\mathbf{z} = \mathbf{s}'_0 + \mathbf{R}^T \cdot \mathbf{s}(\mathbf{v})$, to generate tag ϕ on message m .

Claim 4.17. $\left| \Pr[\mathbf{G}_j = 1] - \Pr[\mathbf{G}'_j = 1] \right| \leq 4\mu\varepsilon$, for all $j \in \{1, \dots, Q_{\text{vrfy}}\}$.

Proof. Fix a value of $j \in \{1, \dots, Q_{\text{vrfy}}\}$. We prove something stronger, namely that the distributions induced by \mathbf{G}_j and \mathbf{G}'_j are computationally close, within distance $4\mu\varepsilon$. Assume the contrapositive, namely that there exists a distinguisher \mathcal{D} that can distinguish games \mathbf{G}_j and \mathbf{G}'_j . We build an attacker \mathcal{B} (running \mathcal{D}) such that

$$\left| \Pr\left[\text{Exp}_{\ell, \mu, n, \tau}^{\text{real}}(\mathcal{B}) = 1\right] - \Pr\left[\text{Exp}_{\ell, \mu, n, \tau}^{\text{rand}}(\mathcal{B}) = 1\right] \right| > 4\mu\varepsilon,$$

contradicting Lemma 4.10. Adversary \mathcal{B} works as follows:

1. At the beginning \mathcal{B} samples h, π (but not \mathbf{S}, s_0, s'_0).
2. Upon input a query m to the tag oracle, \mathcal{B} does the following:
 - Sample a random $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^n$ and compute $\mathbf{v} = h(m, \mathbf{b})$.
 - Query \mathbf{v} to oracle $\text{EVAL}(\cdot)$, obtaining a pair (\mathbf{R}, \mathbf{z}) , and forward $\phi = \pi(\mathbf{R}, \mathbf{z}, \mathbf{b})$ to \mathcal{D} .
3. Upon input a verification query (m, ϕ) to the verification oracle, \mathcal{B} does the following:
 - First check whether (m, ϕ) is equal to one of the tags previously returned to \mathcal{D} ; if this is the case answer with **accept**.
 - Otherwise, check whether (m, ϕ) is the j th verification query; if this is not the case, then answer with **reject**.
 - Else, (m, ϕ) is the j th verification query; call it (m^*, ϕ^*) . Let $(\mathbf{R}^*, \mathbf{z}^*, \mathbf{b}^*) = \pi^{-1}(\phi^*)$, compute $\mathbf{v}^* = h(m^*, \mathbf{b}^*)$ and forward $(\mathbf{R}^*, \mathbf{v}^*)$ to oracle $\text{CHAL}(\cdot)$ obtaining a vector \mathbf{z}' . Check that $\mathbf{wt}(\mathbf{z}' + \mathbf{z}^*) \leq n \cdot \tau'$; if this is the case return **accept** to \mathcal{D} , otherwise return **reject**.
4. Finally \mathcal{B} outputs whatever \mathcal{D} does.

For the analysis, note that \mathcal{B} runs in time similar to that of \mathcal{D} . By inspection, one can verify that in case \mathcal{B} is running in the “real experiment” or in the “random experiment,” the simulation of the tag queries provided by \mathcal{B} is distribute like in \mathbf{G}_j or in \mathbf{G}'_j , respectively. Finally, all verification queries before the j th query are either answered with **accept** (in case they are identical to a previously simulated tag), or with **reject** (otherwise); this is consistent with both games \mathbf{G}_j and \mathbf{G}'_j . The j th verification query is fresh by definition and is simulated using the answer from the $\text{CHAL}(\cdot)$ oracle, so it has the right distribution. The claim follows. \square

Claim 4.18. $\Pr[\mathbf{G}'_j = 1] \leq \alpha''_{\tau', n}$, for all $j \in \{1, \dots, Q_{\text{vrfy}}\}$.

Proof of Claim. Fix a value of $j \in \{1, \dots, Q_{\text{vrfy}}\}$. If \mathbf{R}^* does not have full rank then the experiment outputs 0 by definition. So from now we only consider the case where $\text{rank}(\mathbf{R}^*) = n$. In Game \mathbf{G}'_j , the values (\mathbf{R}, \mathbf{z}) the adversary \mathcal{A} obtains from the tag oracle are independent of the secrets (\mathbf{S}, s_0, s'_0) . Since $\mathbf{s}(\mathbf{v}^*)$ is uniformly random and \mathbf{R}^* has full rank, the vector $\mathbf{x} := s'_0 + \mathbf{R}^{*\text{T}} \cdot \mathbf{s}(\mathbf{v}^*) + \mathbf{z}^*$ is uniformly random over \mathbb{Z}_2^n . Thus the probability that the second verification $\mathbf{wt}(\mathbf{z}^* + s'_0 + \mathbf{R}^{*\text{T}} \cdot \mathbf{s}(\mathbf{v}^*)) \leq n \cdot \tau'$ passes is $\Pr[\mathbf{wt}(\mathbf{x}) \leq n \cdot \tau'] = \alpha''_{\tau', n} = 2^{-\Theta(n)}$. \square

Summing up, in the case $\Pr[E_{\text{fresh}}] > \varepsilon'/2$ (combining the bounds in Claim 4.17—4.18 together with Eq. (4.7)), we can use \mathcal{A} to break the LPN assumption with advantage $\frac{\varepsilon'}{8\mu Q_{\text{vrfy}}} - Q_{\text{vrfy}} \cdot (\alpha + \alpha''_{\tau', n})$. On the other hand in the case $\Pr[E_{\text{fresh}}] \leq \varepsilon'/2$, we have an advantage as given in Eq. (4.5). Thus

$$\varepsilon = \min \left\{ \varepsilon'/2 - \frac{Q^2}{2\mu}, \frac{\varepsilon'}{8\mu Q_{\text{vrfy}}} - Q_{\text{vrfy}} \cdot (\alpha + \alpha''_{\tau', n}) \right\}.$$

\square

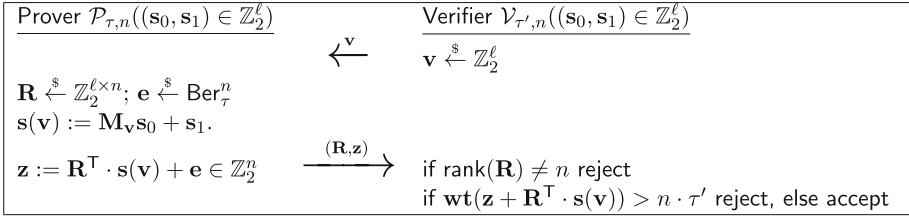


Fig. 9. Two-round protocol AUTH₂ with active security from the LPN assumption.

tation of a finite field multiplication with \mathbf{v} (see definition below), and $(s_0, s_1) \in \mathbb{Z}_2^\ell$ is the secret key.

Definition 4.19. For $\mathbf{c} \in \mathbb{Z}_2^\ell$, let $\mathbf{M}_\mathbf{c} \in \mathbb{Z}_2^{\ell \times \ell}$ denote the matrix of the linear map implementing the finite field multiplication with \mathbf{c} when interpreted as an element in \mathbb{F}_{2^ℓ} .⁹

The statement below follows directly from the properties of finite fields:

$$\text{for all distinct vectors } \mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^\ell, \quad \mathbf{M}_\mathbf{a} - \mathbf{M}_\mathbf{b} \text{ is an invertible matrix.} \quad (4.8)$$

The mapping $\varphi(\mathbf{c}) = \mathbf{M}_\mathbf{c}$ is called encoding with full-rank differences in [10]. An explicit construction was given in [14].

We are now ready to define the modified authentication protocol. Even though the protocol is less efficient than AUTH, it has a considerably simpler proof.

- **Public Parameters.** The authentication protocol has the following public parameters, where τ, τ' are constants and ℓ, n depend on the security parameter λ .

$\ell \in \mathbb{N}$	length of the secret keys $\mathbf{s}_0, \mathbf{s}_1 \in \mathbb{Z}_2^\ell$
$\tau \in]0, 1/2[$	parameter of the Bernoulli error distribution Ber_τ
$\tau' = 1/4 + \tau/2$	acceptance threshold
$n \in \mathbb{N}$	number of parallel repetitions (we require $n \leq \ell/2$)

- **Key Generation.** Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{s}_0, \mathbf{s}_1 \xleftarrow{\$} \mathbb{Z}_2^\ell$ and returns $(\mathbf{s}_0, \mathbf{s}_1)$ as the secret key.
- **Authentication Protocol.** The two-round authentication protocol with prover $\mathcal{P}_{\tau,n}$ and verifier $\mathcal{V}_{\tau',n}$ is given in Fig. 9.

We now sketch the reduction from the $\text{SLPN}_{\tau,\ell,\ell}$ assumption. It is similar to the one of Theorem 3.1, with a slightly different setup of adversary \mathcal{B} . Let \mathbf{s} be the secret of the SLPN oracle. In the reduction, \mathcal{B} first samples a random $\mathbf{v}^* \in \mathbb{Z}_2^\ell$ that will be used as the challenge and implicitly defines the secret key $(\mathbf{s}_0, \mathbf{s}_1)$ as

$$\begin{aligned} \mathbf{s}_0 &:= \mathbf{s} \\ \mathbf{s}_1 &:= -\mathbf{M}_{\mathbf{v}^*} \mathbf{s} + \mathbf{c}, \end{aligned}$$

⁹This representation is unique once the irreducible polynomial f defining $\mathbb{F}_{2^\ell} = \mathbb{F}_2[x]/(f)$ is fixed.

for $\mathbf{c} \xleftarrow{\$} \mathbb{Z}_2^\ell$. This way we have

$$\mathbf{s}(\mathbf{v}) = \begin{cases} (\mathbf{M}_v - \mathbf{M}_{v^*})\mathbf{s} + \mathbf{c} & \mathbf{v} \neq \mathbf{v}^* \\ \mathbf{c} & \mathbf{v} = \mathbf{v}^*, \end{cases}$$

where $\mathbf{M}_v - \mathbf{M}_{v^*}$ is guaranteed to be an invertible matrix by Eq. (4.8). This way, all adversarial queries $\mathbf{v} \neq \mathbf{v}^*$ made in the first phase can be answered by returning (\mathbf{R}, \mathbf{z}) obtained from the SLPN oracle (by calling it with the parameters $\text{SLPN}(\mathbf{M}_v - \mathbf{M}_{v^*}, \mathbf{c})$). As in the proof of Theorem 3.1, the one challenge verification query corresponding to \mathbf{v}^* can be correctly answered with accept or reject as $\mathbf{s}(\mathbf{v}^*)$ does not depend on \mathbf{s} . This way all answers \mathbf{z} in the first phase can be switched from real to random, without the adversary noticing it under the SLPN assumption. Once all answers in the first phase are uniform and independent of the secret key, one can again argue that the adversary has no chance in winning the second phase.

Generalization to LWE

All the protocols presented in this paper are based on the hardness of the LPN problem. A natural generalization of this problem is the learning with errors (LWE) problem [41]. The most appealing characteristic of this problem is that it enjoys for certain parameters a worst-case hardness guarantee [39,41]. We informally recall the LWE problem below. Let $q \geq 2$ be a prime and denote with $\text{Gau}_{q,\tau}$ the so-called discretized normal error distribution parametrized by some $\tau \in]0, 1[$. This distribution is obtained by drawing $x \in \mathbb{R}$ from the Gaussian distribution of width τ (i.e., x is chosen with probability $\frac{1}{\tau} \exp(-\pi x^2/\tau^2)$) and outputting $\lfloor q \cdot x \rfloor \bmod q$. For a random secret $\mathbf{s} \in \mathbb{Z}_q^\ell$, the (decisional) $\text{LWE}_{q,\tau,\ell}$ problem is to distinguish samples of the form $(\mathbf{r}, \mathbf{r}^\top \cdot \mathbf{s} + e)$ from uniformly random samples in $\mathbb{Z}_q^\ell \times \mathbb{Z}_q$, where $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^\ell$, $e \xleftarrow{\$} \text{Gau}_{q,\tau}$ and all the operations are performed modulo q . The subspace/subset version of the LWE problem can be defined exactly in the same fashion as for LPN (cf. Definition 2.2). It was showed in [40] that the subspace/subset LWE problems are equivalent to the LWE problem.

All the protocols in this paper can be generalized to \mathbb{Z}_q and proven secure under the hardness of the subset LWE assumption (and hence the standard LWE assumption). This requires us to sample all the elements from \mathbb{Z}_q (instead of \mathbb{Z}_2), replace Ber_τ with $\text{Gau}_{q,\tau}$ and perform all the operations involved modulo q . We need also to specify how to replace the verification steps involving the computation of Hamming weights $\text{wt}(\cdot)$. Given a vector $\mathbf{e} \in \mathbb{Z}_q^n$ sampled from $\text{Gau}_{q,\tau}^n$ (where \mathbf{e} has the form $\mathbf{z} - \mathbf{R}^\top \cdot \mathbf{s}_{\downarrow v} \bmod q$ for an honest execution of the protocol from Sect. 3 or $\mathbf{z} - \mathbf{R}^\top \cdot \mathbf{s}(\mathbf{v}) \bmod q$ for the schemes from Sect. 4), this can be done by checking that the (squared) Euclidean norm of \mathbf{e} , i.e., the quantity $\|\mathbf{e}\|^2 := \sum_{i=1}^n |\mathbf{e}[i]|^2$, does not exceed $n \lfloor \frac{q}{2} \rfloor \cdot \tau'$ (which will happen with overwhelming probability by the standard tail bound on Gaussians).

The change in domain from \mathbb{Z}_2 to \mathbb{Z}_q buys us security based on a different assumption, which is known to be equivalent (for a proper choice of parameters) to the hardness of well-studied (worst-case) lattice problems. This comes at the price of a higher com-

putational complexity, which may be a problem in the context of resource bounded devices.

Appendix 2: A Technical Lemma

Lemma 4.20. For $n, d \in \mathbb{Z}$, let $\Delta(n, d)$ denote the probability that a random matrix in $\mathbb{Z}_2^{(n+d) \times n}$ has rank less than n , then

$$\Delta(n, d) \leq 2^{-d}.$$

Proof. Assume we sample the n columns of a matrix $\mathbf{M} \in \mathbb{Z}_2^{(n+d) \times n}$ one by one. For $i = 1, \dots, n$ let E_i denote the event that the first i columns are linearly independent, then

$$\Pr[\neg E_i | E_{i-1}] = \frac{2^{i-1}}{2^{n+d}} = 2^{i-1-n-d}$$

as $\neg E_i$ happens iff the i th column (sampled uniformly from a space of size 2^{n+d}) falls into the space (of size 2^{i-1}) spanned by the first $i-1$ columns. We get further

$$\Delta(n, d) = \Pr[\neg E_n] = \sum_{i=1}^n \Pr[\neg E_i | E_{i-1}] = \sum_{i=1}^n 2^{i-1-n-d} \leq 2^{-d}.$$

□

References

- [1] S. Agrawal, D. Boneh, X. Boyen, Efficient lattice (H)IBE in the standard model, in *EUROCRYPT 2010*, volume 6110 of LNCS, ed. by H. Gilbert (Springer, May 2010), pp. 553–572
- [2] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide* (SIAM, Philadelphia, 2000)
- [3] E. Berlekamp, R. McEliece, H. van Tilborg, On the inherent intractability of certain coding problems. *IEEE Trans. Inf. Theory*, **24**(3), 384–386 (1978)
- [4] O. Blazy, E. Kiltz, J. Pan, (Hierarchical) identity-based encryption from affine message authentication, in *CRYPTO 2014*, volume 8616 of LNCS, ed. by J.A. Garay, R. Gennaro (Springer, Aug 2014), pp. 408–425
- [5] A. Blum, M.L. Furst, M.J. Kearns, R.J. Lipton, Cryptographic primitives based on hard learning problems, in *CRYPTO'93*, volume 773 of LNCS, ed. by D.R. Stinson (Springer, Aug 1994), pp. 278–291
- [6] A. Blum, A. Kalai, H. Wasserman, Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, **50**(4), 506–519 (2003)
- [7] S. Bogos, F. Tramèr, S. Vaudenay, On solving LPN using BKW and variants—implementation and analysis. *Cryptogr. Commun.* **8**(3), 331–369 (2016)
- [8] S. Bogos, S. Vaudenay, Observations on the LPN solving algorithm from Eurocrypt'16. Cryptology ePrint Archive, Report 2016/437 (2016). <http://eprint.iacr.org/2016/437>
- [9] S. Bogos, S. Vaudenay, Optimization of LPN solving algorithms. Cryptology ePrint Archive, Report 2016/288 (2016). <http://eprint.iacr.org/2016/288>

- [10] X. Boyen, Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more, in *PKC 2010*, volume 6056 of LNCS, ed. by P.Q. Nguyen, D. Pointcheval (Springer, May 2010), pp. 499–517
- [11] J. Bringer, H. Chabanne, E. Dottax, HB^{++} : a lightweight authentication protocol secure against some attacks, in *SecPerU 2006* (IEEE Computer Society, June 2006), pp. 28–33
- [12] D. Cash, E. Kiltz, S. Tessaro, Two-round man-in-the-middle security from LPN, in *TCC 2016-A*, volume 9562 of LNCS, ed. by E. Kushilevitz, T. Malkin (Springer, Jan 2016), pp. 225–248
- [13] J. Chen, H. Wee, Fully, (almost) tightly secure IBE and dual system groups, in *CRYPTO 2013*, volume 8043 of LNCS, ed. by R. Canetti, J.A. Garay (Springer, Aug 2013), pp. 435–460
- [14] R. Cramer, I. Damgård, On the amortized complexity of zero-knowledge protocols, in *CRYPTO 2009*, volume 5677 of LNCS, ed. by S. Halevi (Springer, Aug 2009), pp. 177–191
- [15] Y. Dodis, E. Kiltz, K. Pietrzak, D. Wichs, Message authentication, revisited, in *EUROCRYPT 2012*, volume 7237 of LNCS, ed. by D. Pointcheval, T. Johansson (Springer, April 2012), pp. 355–374
- [16] D.N. Duc, K. Kim, Securing HB^+ against GRS man-in-the-middle attack, in *2007 symposium on cryptography and information security*, Jan 2007
- [17] J.-B. Fischer, J. Stern, An efficient pseudo-random generator provably as secure as syndrome decoding, in *EUROCRYPT'96*, volume 1070 of LNCS, ed. by U.M. Maurer (Springer, May 1996), pp. 245–255
- [18] M. Fürer, Faster integer multiplication. *SIAM J. Comput.* **39**(3), 979–1005 (2009)
- [19] L. Gaspar, G. Leurent, F.-X. Standaert, Hardware implementation and side-channel analysis of Lapin, in *CT-RSA 2014*, LNCS (Springer, 2014), pp. 206–226
- [20] H. Gilbert, M. Robshaw, H. Sibert, An active attack against HB^+ —a provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237 (2005). <http://eprint.iacr.org/>
- [21] H. Gilbert, M.J.B. Robshaw, Y. Seurin, Good variants of HB^+ are hard to find, in *FC 2008*, volume 5143 of LNCS, ed. by G. Tsudik (Springer, Jan 2008), pp. 156–170
- [22] H. Gilbert, M.J.B. Robshaw, Y. Seurin, HB^{\sharp} : increasing the security and efficiency of HB^+ , in *EUROCRYPT 2008*, volume 4965 of LNCS, ed. by N.P. Smart (Springer, April 2008), pp. 361–378
- [23] O. Goldreich, S. Goldwasser, S. Micali, How to construct random functions. *J. ACM***33**(4), 792–807 (1986)
- [24] Q. Guo, T. Johansson, C. Löndahl, Solving LPN using covering codes, in *ASIACRYPT 2014*, volume 8873 of LNCS, ed. by P. Sarkar, T. Iwata (Springer, Dec 2014), pp. 1–20
- [25] S. Heyse, E. Kiltz, V. Lyubashevsky, C. Paar, K. Pietrzak, Lapin: an efficient authentication protocol based on Ring-LPN, in *FSE 2012*, volume 7549 of LNCS, ed. by A. Canteaut (Springer, March 2012), pp. 346–365
- [26] W. Hoeffding, Probability inequalities for sums of bounded random variables. *J. Am. Stat. Assoc.* **58**(301), 13–30 (1963)
- [27] N.J. Hopper, M. Blum, Secure human identification protocols, in *ASIACRYPT 2001*, volume 2248 of LNCS, ed. by C. Boyd (Springer, Dec 2001), pp. 52–66
- [28] A. Juels, S.A. Weis, Authenticating pervasive devices with human protocols, in *CRYPTO 2005*, volume 3621 of LNCS, ed. by V. Shoup (Springer, Aug 2005), pp. 293–308
- [29] T. Kailath, A.H. Sayed, *Fast Reliable Algorithms for Matrices with Structure* (SIAM, Philadelphia, 1999)
- [30] J. Katz, J.S. Shin, Parallel and concurrent security of the HB and HB^+ protocols, in *EUROCRYPT 2006*, volume 4004 of LNCS, ed. by S. Vaudenay (Springer, May/June 2006), pp. 73–87
- [31] J. Katz, J.S. Shin, A. Smith, Parallel and concurrent security of the HB and HB^+ protocols. *J. Cryptol.* **23**(3), 402–421 (2010)
- [32] M.J. Kearns, Efficient noise-tolerant learning from statistical queries. *J. ACM***45**(6), 983–1006 (1998)
- [33] E. Kiltz, K. Pietrzak, D. Cash, A. Jain, D. Venturi, Efficient authentication from hard learning problems, in *EUROCRYPT 2011*, volume 6632 of LNCS, ed. by K.G. Paterson (Springer, May 2011), pp. 7–26.
- [34] É. Levieil, P.-A. Fouque, An improved LPN algorithm, in *SCN 06*, volume 4116 of LNCS, ed. by R. De Prisco, M. Yung (Springer, Sept 2006), pp. 348–359
- [35] V. Lyubashevsky, D. Masny, Man-in-the-middle secure authentication schemes from LPN and weak PRFs, in *CRYPTO 2013*, volume 8043 of LNCS, ed. by R. Canetti, J.A. Garay (Springer, Aug 2013), pp. 308–325
- [36] V. Lyubashevsky, C. Peikert, O. Regev, On ideal lattices and learning with errors over rings, in *EUROCRYPT 2010*, volume 6110 of LNCS, ed. by H. Gilbert (Springer, June 2010), pp. 1–23

- [37] J. Munilla, A. Peinado, **HB – MP**: a further step in the HB-family of lightweight authentication protocols. *Comput. Netw.* **51**(9), 2262–2267 (2007)
- [38] K. Ouafi, R. Overbeck, S. Vaudenay, On the security of HB# against a man-in-the-middle attack, in *ASIACRYPT 2008*, volume 5350 of LNCS, ed. by J. Pieprzyk (Springer, Dec 2008), pp. 108–124
- [39] C. Peikert, Public-key cryptosystems from the worst-case shortest vector problem: extended abstract, in *41st ACM STOC*, ed. by M. Mitzenmacher (ACM Press, May/June 2009), pp. 333–342
- [40] K. Pietrzak, Subspace LWE, in *TCC 2012*, volume 7194 of LNCS, ed. by R. Cramer (Springer, March 2012), pp. 548–563
- [41] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, in *37th ACM STOC*, ed. by H.N. Gabow, R. Fagin (ACM Press, May 2005), pp. 84–93
- [42] Schönhage, V. Strassen, Schnelle multiplikation grosser zahlen. *Computing* **7**, 281–292 (1971)
- [43] J. Van De Graaf, *Towards a formal definition of security for quantum protocols*. PhD thesis, Universite de Montreal, Monreal, P.Q., Canada, Canada, AAINQ35648, 1998
- [44] B.R. Waters, Efficient identity-based encryption without random oracles, in *EUROCRYPT 2005*, volume 3494 of LNCS, ed. by R. Cramer (Springer, May 2005), pp. 114–127
- [45] J. Watrous, Zero-knowledge against quantum attacks. *SIAM J. Comput.* **39**(1), 25–58 (2009)
- [46] B. Zhang, L. Jiao, M. Wang, Faster algorithms for solving LPN, in *EUROCRYPT 2016*, volume 9665 of LNCS, ed. by M. Fischlin, J.-S. Coron (Springer, May 2016), pp. 168–195