An extended abstract of this paper is published in the proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques—Eurocrypt 2011 [26]. This is the full version.

Efficient Authentication from Hard Learning Problems

Eike Kiltz[*] — Ruhr-Universität Bochum
Krzysztof Pietrzak[†] — IST Austria
David Cash — Rutgers University
Abhishek Jain — Boston University and MIT
Daniele Venturi — Sapienza University of Rome

August 25, 2014

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 (MIM) attacks. A MAC implies such a (two-round) protocol.

Contents

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

[†]Supported by the European Research Council, ERC Starting Grant (259668-PSPC).

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 *learning parity with noise* (LPN) problem. 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 [19] (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 [2, 4, 6, 25, 21, 34]. It is closely related to the well-studied problem of decoding random linear codes, and unlike most number-theoretic 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 [21] suggested that humans could perform the computation in their provably-secure scheme, even with realistic parameters. The efficiency of LPN-based schemes also makes them suitable for weak devices like RFID tags, where even evaluating a blockcipher may be prohibitive.

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 $\mathcal{V}$ (in the context of RFID implementations, we think of $\mathcal{P}$ as the "tag" and $\mathcal{V}$ as the "reader"). 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) in the first phase.

The LPN problem. Briefly stated, the LPN problem is to distinguish from random several "noisy inner products" of random binary vectors with a random secret vector.

More formally, 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 $(\mathbf{r}, \mathbf{r}^\mathsf{T}\mathbf{x} + e)$, where $\mathbf{r} \in \mathbb{Z}_2^\ell$ is uniformly random and $e \in \mathbb{Z}_2$ is selected according to Ber_τ, the Bernoulli distribution over $\mathbb{Z}_2$ with parameter τ (i.e., $\Pr[e = 1] = \tau$). The $\mathsf{LPN}_{\tau,\ell}$ problem is to

$\underline{\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^\ell)}$		$\underline{\mathcal{V}_{\tau',n}(\mathbf{s} \in \mathbb{Z}_2^\ell)}$
	$\xleftarrow{\mathbf{R}}$	$\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$
$\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$		
$\mathbf{z} := \mathbf{R}^\mathsf{T} \cdot \mathbf{s} + \mathbf{e}$	$\xrightarrow{z}$	verify: $\mathbf{wt}(\mathbf{z} + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}) < \tau' \cdot n$

Figure 1: The HB protocol, secure against passive attacks.

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.* [4] that this is equivalent to the search version of LPN, where one needs to compute $\mathbf{x}$ given oracle access to $\Lambda_{\tau,\ell}(\mathbf{x})$ (cf. [24, Thm.2] for precise bounds). We note that the search and decision variants are solvable with a linear in ℓ number of samples when there is no noise, i.e. when $\tau = 0$, and the best algorithms take time $2^{\ell/\log \ell}$ when $\tau > 0$ is treated as a constant [5, 6, 27].

Authentication protocols from LPN. Starting with the work of Hopper and Blum [21], 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 $z = \mathbf{r}^\mathsf{T}\mathbf{s} + e$, where $e \in \mathbb{Z}_2$ is sampled according to Ber_τ. Finally, the verifier accepts if $z = \mathbf{r}^\mathsf{T}\mathbf{s}$.

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 $\mathbf{r}^\mathsf{T}\mathbf{s} = z$ with probability $1/2$). This can be reduced via sequential or parallel composition. The parallel variant, denoted HB, is illustrated in Figure 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 are correct.

The 2-round HB protocol is provably secure against passive attacks, but efficient active attacks are known against it. This is unsatisfying because in several scenarios, and especially in RFID applications, an adversary *will* be able to mount an active attack. Subsequently, Juels and Weis [22] proposed an efficient 3 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 [23, 24]. The protocol (in its parallel repetition variant) is illustrated in Figure 2.

Gilbert *et al.* [16] showed that HB^+ can be broken by a MIM attack. Several variants HB^{++} [9], HB^* [13], HB-MP [30] were proposed to prevent the particular attack from [16], but all of them were later shown to be insecure [17]. In [18], a variant $\mathsf{HB}^\#$ was presented which provably resists the particular attack from [16], but was shown susceptible to a more general MIM attack [31]. However, no improvements in terms of round complexity, security or tightness of the reduction over HB^+ were achieved: 3 round protocols achieving active security $\sqrt{\varepsilon}$ (assuming LPN is ε-hard) are the state of the art.

$\underline{\mathcal{P}_{\tau,n}(\mathbf{s}_1,\mathbf{s}_2)}$		$\underline{\mathcal{V}_{\tau',n}(\mathbf{s}_1,\mathbf{s}_2)}$
$\mathbf{R}_1 \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$	$\xrightarrow{\mathbf{R}_1}$	
	$\xleftarrow{\mathbf{R}_2}$	$\mathbf{R}_2 \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$
$\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$		
$\mathbf{z} := \mathbf{R}_1^\mathsf{T} \cdot \mathbf{s}_1 + \mathbf{R}_2^\mathsf{T} \cdot \mathbf{s}_2 + \mathbf{e}$	$\xrightarrow{z}$	verify: $\mathbf{wt}(\mathbf{z} + \mathbf{R}_1^\mathsf{T} \cdot \mathbf{s}_1 + \mathbf{R}_2^\mathsf{T} \cdot \mathbf{s}_2) \leq \tau' \cdot n$

Figure 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 [22]) which moreover has a tight security reduction (an open problem mentioned in [24]). 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.

2-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 [21, 22, 24, 18], 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 $\mathbf{R}^\mathsf{T} \cdot \mathbf{s} + \mathbf{e}$, 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 $\mathbf{s} = (1,1,1,0,0,0)^\mathsf{T}, \mathbf{v} = (0,1,1,1,0,0)^\mathsf{T}$ then $\mathbf{s}_{\downarrow \mathbf{v}} = (1,1,0)^\mathsf{T}$.) The prover then picks $\mathbf{R}$ by *itself* and computes noisy inner products of the form $\mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e}$. 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 $\mathbf{R}$ were sent by an active adversary.

Our proof relies on the recently introduced *subspace LPN problem* [33]. In contrast to the active-attack security proof of HB^+ [24], 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 [36], classical proofs using rewinding in general do not translate to the quantum setting (cf. [38] for a more recent discussion).

Let us emphasise 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 & Man-In-The-Middle Security. In Section 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. Note 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 $\phi = (\mathbf{R}, \mathbf{z} = \mathbf{R}^{\mathsf{T}} \cdot f_{\mathbf{s}}(\mathbf{m}) + \mathbf{e})$, where the secret key derivation function $f_{\mathbf{s}}(\mathbf{m}) \in \mathbb{Z}_2^{\ell}$ first uniquely encodes the message $\mathbf{m}$ 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 ith, then the ith bit of $f_{\mathbf{s}}(\mathbf{m})$ must be 1.[1] Our solution is to randomize the mapping f, i.e. use $f_{\mathbf{s}}(\mathbf{m}, \mathbf{b})$ for some randomness $\mathbf{b}$ and compute the tag as $\phi = \pi(\mathbf{R}, \mathbf{R}^{\mathsf{T}} \cdot f_{\mathbf{s}}(\mathbf{m}, \mathbf{b}) + \mathbf{e}, \mathbf{b})$, 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 since it has to guess the value $\mathbf{v}$ from the adversary's forgery.[2] In our case, however, this still leads to a polynomial security reduction when one commits to the hardness of the LPN problem at the time of the construction. (See the first paragraph of §4 for a discussion.)

To get a strictly polynomial security reduction (without having to commit to the hardness of the LPN problem), in our second construction we instantiate the above MAC with a different secret key derivation function $f_{\mathbf{s}}(\mathbf{m}, \mathbf{b}) = \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$ (where $\mathbf{v} = h(\mathbf{m}, \mathbf{b})$ 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 $\mathbf{S} \in \mathbb{Z}_2^{\ell \times \mu}$. Our security reduction uses a technique from [10, 3].[3]

1.2 Efficiency

Figure 3 gives a rough comparison of our new protocol and MACs with the $\mathsf{HB}, \mathsf{HB}^+$ protocols and, as a reference, also the classical tree-based GGM construction [19]. The second row in the table specifies the security notion that is (provably) achieved under the $\mathsf{LPN}_{\tau,\ell}$ assumption. λ is a security parameter and n denotes the number of "repetitions". Typical parameters can be $\ell = 500, \lambda = 80, n = 250$. Computation complexity counts the number of binary operations over $\mathbb{F}_2$. Communication complexity counts the total length of all exchanged messages.[4] The last row in the table states the tightness of the security reduction, i.e. what exact security is achieved (ignoring constants and higher order terms) assuming the $\mathsf{LPN}_{\tau,\ell}$ problem is ε-hard.

[1] In fact, 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.

[2] In the context of identity-based encryption (IBE) a similar idea has been used to go from selective-ID to full security using "complexity leveraging" [7].

[3] An earlier version of this paper adapted a technique originally used by Waters [37] in the context of IBE schemes that has been applied to lattice based signature [8] and encryption schemes [1].

[4] For MACs, we consider the communication one incurs by constructing a MIM secure 2-round protocol from the MAC by having the prover compute the tag on a random challenge message.

Construction	Security	Complexity		Key-size	Reduction
		Communication	Computation		
HB [21]	passive (2 rnd)	$\ell \cdot n/c$	$\Theta(\ell \cdot n)$	$\ell \cdot c$	ε (tight)
HB^+ [22]	active (3 rnd)	$\ell \cdot n \cdot 2/$ c	$\Theta(\ell \cdot n)$	$\ell \cdot 2 \cdot c$	$\sqrt{\varepsilon}$
AUTH § 3	active (2 rnd)	$\ell \cdot n \cdot 2.1/c$	$\Theta(\ell \cdot n)$	$\ell \cdot 4.2 \cdot c$	ε (tight)
MAC_1 § 4.1	MAC → MIM (2 rnd)	$\ell \cdot n \cdot 2.1/c$	$\Theta(\ell \cdot n) + \mathsf{PIP}$	$\ell \cdot 12.6 \cdot c$	$\sqrt{\varepsilon} \cdot Q$ ($\star$)
MAC_2 § 4.2	MAC → MIM (2 rnd)	$\ell \cdot n \cdot 1.1/c$	$\Theta(\ell \cdot n) + \mathsf{PIP}$	$\ell \cdot \lambda \cdot c$	$\varepsilon \cdot Q$
GGM [19]	PRF → MIM (2 rnd)	λ	$\Theta(\ell^2 \cdot \lambda)$	$\Theta(\ell)$	$\varepsilon \cdot \lambda$

Figure 3: A comparison of our new authentication protocol and MACs with the HB, HB^+ protocols and the classical GGM construction. The trade-off parameter $c, 1 \le c \le n$ and the term PIP will be explained in the "Communication vs. Key-Size" paragraph below. ($\star$) See discussion in §4.

The prover and verifier in the HB, HB^+ and our new protocols have to perform $\Theta(\ell \cdot n)$ basic binary operations, assuming the $\mathsf{LPN}_{\tau,\ell}$ 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 $\mathsf{LPN}_{\tau,\ell}$ assumption with $\Theta(\ell^2)$ binary operations [14]. Via the classical GGM construction [19], 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. (Recall that $\ell \approx 500$.)

Communication vs. Key-Size. For all constructions except GGM, there is a natural trade-off between communication and key-size, where for any constant c ($1 \le c \le n$), 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 our MACs: to compute or verify a tag one has to evaluate a pairwise independent permutation (PIP) on the entire tag of length $m := \Theta(\ell \cdot n/c)$.

The standard way to construct a PIP π over $\mathbb{Z}_{2^m}$ is to define $\pi(x) := a \cdot x + b \in \mathbb{F}_{2^m}$ for random $a, b \in \mathbb{F}_{2^m}$. Thus the computational cost of evaluating the PIP is one multiplication of two m bits values: the PIP term in the table accounts for this complexity. Asymptotically, such a multiplication takes only $O(m \log m \log\log m)$ time [35, 15], but for small m (like in our scheme) this will not be faster than using schoolbook multiplication, which takes $\Theta(m^2)$ time. For parameters $\ell = 500, n = 250$ and trade-off $c = n$ (which minimizes the tag-length m) we get $m \approx 1200$ for MAC_1 (i.e., $1200 = 2\ell$ plus some statistical security parameters) and $m \approx 600$ for MAC_2. Hence, depending on the parameters, the evaluation of the PIP may be the computational bottleneck of our MACs.

1.3 Subsequent Work

The results obtained in this paper have recently served as a basis for other works on efficient authentication. In [12], 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.* [20] considered a variant of our actively secure authentication protocol based on the Ring LWE problem (introduced in [29]).

Lyubashevsky and Masny [28] built authentication protocols from LPN that are man-in-middle secure for sequential sessions.

2 Definitions

2.1 Notation

For a positive integer k, $[k]$ denotes the set $\{1, \ldots, k\}$; $[0]$ 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 $i \in \{1, \ldots, |\mathbf{x}|\}$ where $\mathbf{x}[i] \neq 0$. For $\mathbf{v} \in \mathbb{Z}_2^m$ we denote by $\overline{\mathbf{v}}$ its inverse, i.e. $\overline{\mathbf{v}}[i] = 1 - \mathbf{v}[i]$ 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 ith 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 $\mathbf{x} \wedge \mathbf{v} = \mathbf{z} \in \mathbb{Z}_2^m$ with $\mathbf{z}[i] = \mathbf{x}[i] \wedge \mathbf{y}[i]$ and $\mathbf{x} \vee \mathbf{v} = \mathbf{z} \in \mathbb{Z}_2^m$ 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 in a passive attack, but in the first phase, the adversary $\mathcal{A}$ is

[5] By 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 looses a factor of k in the security reduction.

$\underline{\mathbf{Exp}^{\mathrm{uf-cma}}_{\mathsf{MAC}}(\mathcal{A}, \lambda)}$

$\mathcal{Q}_{\mathsf{tag}}, \mathcal{Q}_{\mathsf{vrfy}} := \emptyset$

$K \leftarrow \mathcal{K}(1^\lambda)$

$\mathcal{A}^{\mathrm{TAG}(\cdot), \mathrm{VER}(\cdot,\cdot)}(1^\lambda)$

$$\text{Return} \begin{pmatrix} \exists (\mathbf{m}, \phi) \in \mathcal{Q}_{\mathsf{vrfy}} \text{ s.t. } \mathbf{m} \notin \mathcal{Q}_{\mathsf{tag}} \\ \wedge \mathcal{V}(K, \mathbf{m}, \phi) = \mathsf{accept} \end{pmatrix}$$

$\underline{\text{Oracle } \mathrm{TAG}(\mathbf{m})}$

$\mathcal{Q}_{\mathsf{tag}} := \mathcal{Q}_{\mathsf{tag}} \cup \{\mathbf{m}\}$

$\phi \leftarrow \mathcal{T}(K, \mathbf{m})$

Return ϕ

$\underline{\text{Oracle } \mathrm{VER}(\mathbf{m}, \phi)}$

$\mathcal{Q}_{\mathsf{vrfy}} := \mathcal{Q}_{\mathsf{vrfy}} \cup \{(\mathbf{m}, \phi)\}$

Return $\mathcal{V}(K, \mathbf{m}, \phi)$

Figure 4: Experiment $\mathbf{Exp}^{\mathrm{uf-cma}}_{\mathsf{MAC}}(\mathcal{A}, \lambda)$ defining uf-cma security of MAC.

additionally given access to $\mathcal{P}(\mathbf{x})$. For our 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. The second phase is the same as in passive and active attack. In the first phase, the adversary interact (concurrently) with any number of provers and – unlike in an active attacks – also verifiers. The adversary gets to learn the verifiers $\mathsf{accept}/\mathsf{reject}$ decisions. 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 $\mathsf{MAC} = \{\mathcal{K}, \mathcal{T}, \mathcal{V}\}$ 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 $\mathbf{m} \in \mathcal{M}$ 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 $\mathbf{m} \in \mathcal{M}$ and a tag $\phi \in \mathcal{T}$ and outputs $\{\mathsf{accept}, \mathsf{reject}\}$.

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 $\mathcal{V}(K, \mathbf{m}, \phi) = \mathsf{accept}$ iff $\mathcal{T}(K, \mathbf{m}) = \phi$.

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

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

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 $\mathbf{Exp}^{\mathrm{uf-cma}}_{\mathsf{MAC}}(\mathcal{A}, \lambda)$ of Figure 4 outputs 1. The experiment features an adversary $\mathcal{A}$ that issues tag queries on messages $\mathbf{m}$, and verification queries on pairs $(\mathbf{m}, \phi)$; the adversary is successful if she ever asks a verification query $(\mathbf{m}, \phi)$ that is accepted, for some message $\mathbf{m}$ 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 $\Pr[\mathbf{Exp}^{\mathrm{uf-cma}}_{\mathsf{MAC}}(\mathcal{A}, \lambda) = 1] \leq \varepsilon$.

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 $x \leftarrow \mathsf{Ber}_\tau$). For $\ell \geq 1$, Ber^ℓ_τ denotes the distribution over $\mathbb{Z}^\ell_2$ where each vector consists of ℓ independent samples drawn from Ber_τ. Given a secret $\mathbf{x} \in \mathbb{Z}^\ell_2$ and $\tau \in]0, \frac{1}{2}[$, we write $\Lambda_{\tau,\ell}(\mathbf{x})$ for the distribution over $\mathbb{Z}^\ell_2 \times \mathbb{Z}_2$ whose samples are obtained by sampling vectors $\mathbf{r} \stackrel{\$}{\leftarrow} \mathbb{Z}^\ell_2$ and $e \stackrel{\$}{\leftarrow} \mathsf{Ber}_\tau$ outputting $(\mathbf{r}, \mathbf{r}^\mathsf{T} \cdot \mathbf{x} + e)$.

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 $\mathbf{x} \in \mathbb{Z}^\ell_2$) from the uniform distribution on $\ell + 1$ bits denoted $U_{\ell+1}$.

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

$$\left|\Pr\left[\mathcal{D}^{\Lambda_{\tau,\ell}(\mathbf{x})} = 1;\ \mathbf{x} \stackrel{\$}{\leftarrow} \mathbb{Z}^\ell_2\right] - \Pr\left[\mathcal{D}^{U_{\ell+1}} = 1\right]\right| \leq \varepsilon.$$

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) recently introduced in [33]. 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 $\mathbf{x} \in \mathbb{Z}^\ell_2$ and any $\mathbf{A} \in \mathbb{Z}^{\ell\times\ell}_2, \mathbf{b} \in \mathbb{Z}^\ell_2$ we define the distribution

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

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) $\mathsf{SLPN}_{\tau,\ell,d}$ problem is (t, Q, ε)-hard if for every distinguisher $\mathcal{D}$ running in time t and making Q queries,

$$\left|\Pr\left[\mathcal{D}^{\Gamma_{\tau,\ell,d}(\mathbf{x},\cdot,\cdot)} = 1;\ \mathbf{x} \stackrel{\$}{\leftarrow} \mathbb{Z}^\ell_2\right] - \Pr\left[\mathcal{D}^{U_{\ell+1}(\cdot,\cdot)} = 1\right]\right| \leq \varepsilon,$$

where $U_{\ell+1}(\cdot, \cdot)$ on input $\mathbf{A}, \mathbf{b}$ outputs a sample of $U_{\ell+1}$ if $\mathsf{rank}(\mathbf{A}) \geq d$ 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. In [33] a more general statement is proven where one considers the more general Learning With Errors (LWE) assumption,[6] and allows the distinguisher to also apply an affine function to the randomness $\mathbf{r}$ (not just $\mathbf{x}$ as discussed here). For completeness we include the proof for the case of subspace LPN.

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

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

Proof. Let $\mathcal{D}'$ be an adversary with advantage ε' for the $\mathsf{SLPN}_{\tau,\ell,d+g}$ problem, from this $\mathcal{D}'$ we will construct an $\mathcal{D}$ with advantage $\varepsilon \geq \varepsilon' - Q/2^g$ for the $\mathsf{LPN}_{\tau,d}$ 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 $(\hat{\mathbf{r}}, \hat{\mathbf{z}}) \mapsto (\hat{\mathbf{r}}, \hat{\mathbf{z}} + \hat{\mathbf{r}}^\mathsf{T}\mathbf{b})$ reduces queries with arbitrary $\mathbf{b}$ to queries with $\mathbf{b} = \mathbf{0}$.

Our $\mathcal{D}$ will transform the samples of the form $(\mathbf{r}, \mathbf{r}^\mathsf{T}\mathbf{x} + e)$ it gets (where e is either sampled according to Ber_τ or uniform) into samples $(\hat{\mathbf{r}}, \hat{\mathbf{r}}^\mathsf{T}\mathbf{A}\hat{\mathbf{x}}+e)$ for any $\mathbf{A} \in \mathbb{Z}_2^{\ell\times\ell}$ of rank $\geq d$. 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 $\mathbf{W} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell\times d}, \mathbf{w} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\ell$ 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 $\mathsf{rank}(\mathbf{A}) < d + g$ return $\perp$ to $\mathcal{D}'$. Otherwise, query the oracle $\Lambda_{\delta,d}(\mathbf{x})$ to get a sample $(\mathbf{r}, \mathbf{r}^\mathsf{T} \cdot \mathbf{x} + e)$. Define the set $\mathcal{S} \subseteq \mathbb{Z}_2^\ell$ of solutions to the system of linear equations:

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

If $\mathbf{AW}$ has rank $\geq d$ then $\mathcal{S}$ is non-empty (if the rank is d, then the system has exactly one solution, if it has rank $> d$, the system is under-defined and thus has several solutions). $\mathcal{D}$ samples $\hat{\mathbf{r}} \stackrel{\$}{\leftarrow} \mathcal{S}$ and outputs the sample

$$(\hat{\mathbf{r}}, \mathbf{r}^\mathsf{T}\mathbf{x} + \hat{\mathbf{r}}^\mathsf{T}\mathbf{A}\mathbf{w} + e), \tag{2.1}$$

For the analysis, note that $\mathcal{D}$ runs in time $t \approx t'$. It remains to show that simulation performed by $\mathcal{D}$ is correct. This is shown in the following claims.

Claim 2.4. *If $\mathbf{V} = \mathbf{AW}$ has rank $\geq d$, then $\hat{\mathbf{r}} \stackrel{\$}{\leftarrow} \mathcal{S}$ 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, $\Pr[\mathbf{vAW} = \mathbf{r}^\mathsf{T}] = 2^{-d}$, 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 $\Pr[\mathbf{v} = \hat{\mathbf{r}} \mid \mathbf{vAW} = \mathbf{r}^\mathsf{T}] = 2^{d-\ell}$. We get

$$\Pr[\hat{\mathbf{r}} = \mathbf{v} \mid \mathbf{W}, \mathbf{A}, \mathbf{w}] = \Pr[\mathbf{vAW} = \mathbf{r}^\mathsf{T}] \Pr[\mathbf{v} = \hat{\mathbf{r}} \mid \mathbf{vAW} = \mathbf{r}^\mathsf{T}] = 2^{-d}2^{d-\ell} = 2^{-\ell}$$

□

Claim 2.5. *$\mathcal{D}$ perfectly simulates the distribution $\Gamma_{\delta,\ell,d+g}(\hat{\mathbf{x}}, \mathbf{A})$ (where $\hat{\mathbf{x}} = \mathbf{W}\mathbf{x} + \mathbf{w}$).*

[6] See also Appendix A.3 for a discussion.

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

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

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

Claim 2.6. *With probability at least 2^{-g} the set $\mathcal{S}$ is non-empty.*

Proof of Claim. Recall that the set $\mathcal{S}$ is empty 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 $\mathsf{rank}(\mathbf{A}) \geq d + g$ and $\mathbf{W} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell\times d}$.

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 B.1, 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 $\mathsf{rank}(\mathbf{A}\mathbf{W}) < d$). 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. □

2.4.3 Subset Learning Parity with Noise

For some of our constructions, we will only need a weaker version of the $\mathsf{SLPN}_{\tau,\ell,d}$ problem that we call sub**set** 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 $\mathsf{diag}(\mathbf{v}) \in \mathbb{Z}_2^{\ell\times\ell}$ denote the zero matrix with $\mathbf{v}$ in the diagonal, and let

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

Definition 2.7 (Subset LPN)**.** Let $\ell, d \in \mathbb{Z}$ where $d \leq \ell$. The $\mathsf{SLPN}^*_{\tau,\ell,d}$ problem is (t, Q, ε)-hard if for every distinguisher $\mathcal{D}$ running in time t and making Q queries,

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

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

Remark 2.8. $\Gamma^*_{\tau,\ell,d}(\mathbf{x},\mathbf{v})$ samples are of the form $(\mathbf{r}, \mathbf{r}_{\downarrow\mathbf{v}}^{\mathsf{T}}\mathbf{x}_{\downarrow\mathbf{v}} + e) \in \mathbb{Z}_2^{\ell+1}$, where $e \stackrel{\$}{\leftarrow} \mathsf{Ber}_\tau$. To compute the inner product only $\mathbf{r}_{\downarrow\mathbf{v}} \in \mathbb{Z}_2^{\mathbf{wt}(\mathbf{v})}$ is needed, the remaining bits $\mathbf{r}_{\downarrow\overline{\mathbf{v}}} \in \mathbb{Z}_2^{\ell-\mathbf{wt}(\mathbf{v})}$ 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 $(\mathbf{r}_{\downarrow\mathbf{v}}, \mathbf{r}_{\downarrow\mathbf{v}}^{\mathsf{T}}\mathbf{x}_{\downarrow\mathbf{v}} + e) \in \mathbb{Z}_2^{\mathbf{wt}(\mathbf{v})+1}$.

3 Two-Round Authentication with Active Security

In this section we describe our new 2-round authentication protocol and prove its active security under the hardness of the $\mathsf{SLPN}^*_{\tau,2\ell,d}$ problem, where $d = \ell/(2+\gamma)$ for some constant $\gamma > 0$. (Concretely, $\gamma = 0.1$ should do for all practical purposes.)

- 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 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 $n \leq \ell/2$)
- Key Generation. Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{s} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{2\ell}$ and returns $\mathbf{s}$ as the secret key.
- Authentication Protocol. The 2-round authentication protocol with prover $\mathcal{P}_{\tau,n}$ and verifier $\mathcal{V}_{\tau',n}$ is given in Figure 5.

Prover $\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$		Verifier $\mathcal{V}_{\tau',n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell})$
	$\xleftarrow{\mathbf{v}}$	$\mathbf{v} \stackrel{\$}{\leftarrow} \{\mathbf{x} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{x}) = \ell\}$
if $\mathbf{wt}(\mathbf{v}) \neq \ell$ abort		
$\mathbf{R} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell \times n}$; $\mathbf{e} \stackrel{\$}{\leftarrow} \mathsf{Ber}_\tau^n$		
$\mathbf{z} := \mathbf{R}^{\mathsf{T}} \cdot \mathbf{s}_{\downarrow\mathbf{v}} + \mathbf{e} \in \mathbb{Z}_2^n$	$\xrightarrow{(\mathbf{R},\mathbf{z})}$	if $\mathsf{rank}(\mathbf{R}) \neq n$ reject
		if $\mathbf{wt}(\mathbf{z} + \mathbf{R}^{\mathsf{T}} \cdot \mathbf{s}_{\downarrow\mathbf{v}}) > n \cdot \tau'$ reject, else accept

Figure 5: Two-round protocol AUTH with active security from the LPN assumption.

Theorem 3.1. *For any constant $\gamma > 0$, let $d = \ell/(2+\gamma)$. If the $\mathsf{SLPN}^*_{\tau,2\ell,d}$ problem is (t, nQ, ε)-hard then the authentication protocol from Figure 5 is (t', Q, ε')-secure against active adversaries, where for constants $c_\gamma, c_\tau > 0$ that depend only on γ and τ respectively,*

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

The protocol has completeness error $2^{-c'_\tau \cdot n}$ where $c'_\tau > 0$ depends only on τ.

3.1 Proof of completeness

For any $n \in \mathbb{N}$, $\tau \in]0, 1/2[$, let

$$\alpha_{\tau,n} := \Pr[\mathbf{wt}(\mathbf{e}) > n \cdot \tau';\ \mathbf{e} \stackrel{\$}{\leftarrow} \mathsf{Ber}_\tau^n] = 2^{-c''_\tau \cdot n} \tag{3.1}$$

denote the probability that n independent Bernoulli samples with bias τ contain more than a $\tau' := 1/4 + \tau/2$ fraction of 1's. The last equality in Eq. (3.1) follows from the Hoeffding bound, where the constant $c''_\tau > 0$ depends only on τ.

We now prove that the authentication protocol has completeness error $\alpha \leq 2^{-\ell+n} + \alpha_{\tau,n}$. 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. It is easy to show that the probability of this event is $\leq 2^{-n}$ (cf. Lemma B.1). Now, let $\mathbf{e} := \mathbf{z} + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow \mathbf{v}}$ denote the noise added by $\mathcal{P}_{\tau,n}$. Then, in the second verification step, the verifier rejects if $\mathbf{wt}(\mathbf{e}) > n \cdot \tau'$. From Eq. (3.1), we have that this happens with probability $\alpha_{\tau,n}$. This completes the proof of completeness.

3.2 Proof of security

We first define some 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. Using the fact that the expected Hamming weight is $\ell/2 = d(1+\gamma/2) = d(1+\Theta(1))$, one can show that there exists a constant $c_\gamma > 0$ (only depending on γ), such that

$$\alpha'_{\ell,d} := \frac{\sum_{i=0}^{d-1} \binom{\ell}{i}\binom{\ell}{\ell-i}}{\binom{2\ell}{\ell}} \leq 2^{-c_\gamma \cdot \ell}. \tag{3.2}$$

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 $\mathbf{wt}(\mathbf{y}) \leq n \cdot \tau'$. From the Hoeffding bound, it follows that there exists a constant $c_\tau > 0$ (only depending on τ), such that

$$\alpha''_{\tau',n} := 2^{-n} \cdot \sum_{i=0}^{\lfloor n\cdot\tau' \rfloor} \binom{n}{i} \leq 2^{-c_\tau \cdot n}. \tag{3.3}$$

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 2.7. 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:

$$\Pr[\mathcal{B}^{\Gamma^*_{\tau,2\ell,d}(\mathbf{x},\cdot)} = 1] \geq \varepsilon' - Q \cdot \alpha'_{\ell,d} \qquad \text{and} \qquad \Pr[\mathcal{B}^{U_{2\ell+1}(\cdot)} = 1] \leq \alpha''_{\tau',n} .$$

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

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

$$\mathbf{x}^* \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{2\ell}, \qquad \mathbf{v}^* \stackrel{\$}{\leftarrow} \{\mathbf{y} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{y}) = \ell\}.$$

$\mathcal{B}^\mathcal{O}$ will use $\mathbf{v}^*$ as the verifier message during the final phase of the simulated active security game, and it will arrange so that $\mathbf{x}^*$ is the session key for that phase. 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

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

Thus one half of the bits of $\mathbf{s}$ come from $\mathbf{x}^*$, and the other half come from the unknown secret $\mathbf{x}$, with the positions randomly chosen. Whenever $\mathcal{A}$ outputs ϕ^*, 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 $\mathcal{A}$ outputs a valid forgery.

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 find a valid forgery 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 $\mathcal{A}$. (In what follows we assume $\mathbf{v} \neq \mathbf{v}^*$, otherwise $\mathcal{B}^{\mathcal{O}}$ simply returns $\perp$ to $\mathcal{A}$.) Let

$$\mathbf{u}^* := \mathbf{v} \wedge \mathbf{v}^* \qquad \mathbf{u} := \mathbf{v} \wedge \overline{\mathbf{v}}^*.$$

1. $\mathcal{B}^{\mathcal{O}}$ queries its oracle n times on the input $\mathbf{u}$. If the oracle's output is $\perp$ (which happens iff $\mathbf{wt}(\mathbf{u}) < d$), $\mathcal{B}^{\mathcal{O}}$ outputs 0 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 $\mathbf{z}_0 = \hat{\mathbf{R}}_0^{\mathsf{T}} \cdot (\mathbf{x}^* \wedge \mathbf{u}^*)$.
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 $\hat{\mathbf{R}}_{\downarrow\overline{\mathbf{v}}^*} = \hat{\mathbf{R}}_1$.

Second phase. Eventually, $\mathcal{A}$ enters the second phase of the active attack and outputs a forgery $\phi^* = (\mathbf{R}^*, \mathbf{z}^*)$. Then $\mathcal{B}^{\mathcal{O}}$ checks if

$$\mathsf{rank}(\mathbf{R}^*) = n \quad \text{and} \quad \mathbf{wt}(\mathbf{z}^* + \mathbf{R}^{*\mathsf{T}} \cdot \mathbf{x}^*_{\downarrow\mathbf{v}^*}) \leq n \cdot \tau'. \tag{3.5}$$

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

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

Proof of Claim. If $\mathbf{R}^*$ does not have full rank then $\mathcal{B}$ outputs 0 by definition. Therefore, we now consider the case where $\mathsf{rank}(\mathbf{R}^*) = n$.

The answers $\phi = (\mathbf{R}, \mathbf{z})$ that the adversary $\mathcal{A}$ obtains from $\mathcal{B}^{U_{2\ell+1}(\cdot)}$ 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

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

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

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

Proof of Claim. We split the proof in two parts. First we 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 $\mathbf{wt}(\mathbf{v}) < d$), i.e.,

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

Then we upper bound the gap between the probability that $\mathcal{B}$ outputs 1 in the above case and the probability that $\mathcal{B}$ outputs 1 when given access to the oracle that we are interested in as:

$$\left|\Pr[\mathcal{B}^{\Gamma^*_{\tau,2\ell,d}(\mathbf{x},\cdot)} = 1] - \Pr[\mathcal{B}^{\Gamma^*_{\tau,2\ell,0}(\mathbf{x},\cdot)} = 1]\right| \leq Q \cdot \alpha'_{\ell,d}. \tag{3.7}$$

The claim then follows by the triangle inequality from the two equations above. Eq. (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 $(\hat{\mathbf{R}}, \mathbf{z} = \hat{\mathbf{R}}^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e})$ and then forward the compressed version of this samples to $\mathcal{A}$ (that is, $(\mathbf{R}, \mathbf{z} = \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow\mathbf{v}} + \mathbf{e})$ where $\mathbf{R} = \hat{\mathbf{R}}_{\downarrow\mathbf{v}}$, cf. Remark 2.8). We next show that the $\mathbf{z}$ computed by $\mathcal{B}$ indeed have exactly this distribution. In the first step, $\mathcal{B}$ queries its oracle with $\mathbf{u} = \mathbf{v} \wedge \overline{\mathbf{v}}^*$ 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.,
  $$\mathbf{z}_1 = \hat{\mathbf{R}}_1^\mathsf{T} \cdot (\mathbf{x} \wedge \mathbf{u}) + \mathbf{e} = \hat{\mathbf{R}}_1^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{u}) + \mathbf{e}.$$
  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.,
  $$\mathbf{z}_0 = \hat{\mathbf{R}}_0^\mathsf{T} \cdot (\mathbf{x}^* \wedge \mathbf{u}^*) = \hat{\mathbf{R}}_0^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{u}^*).$$
  In the third step, $\mathcal{B}$ then generates $(\hat{\mathbf{R}}, \hat{\mathbf{R}}^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e})$ from the previous values where $\hat{\mathbf{R}}$ is defined by $\hat{\mathbf{R}}_{\downarrow\mathbf{v}^*} = \hat{\mathbf{R}}_0$ and $\hat{\mathbf{R}}_{\downarrow\overline{\mathbf{v}}^*} = \hat{\mathbf{R}}_1$. Using $\mathbf{v} = \mathbf{u} + \mathbf{u}^*$, we get
  $$\begin{aligned} \mathbf{z} &= \mathbf{z}_0 + \mathbf{z}_1 \\ &= \hat{\mathbf{R}}_0^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{u}^*) + \hat{\mathbf{R}}_1^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{u}) + \mathbf{e} \\ &= \hat{\mathbf{R}}^\mathsf{T} \cdot (\mathbf{s} \wedge \mathbf{v}) + \mathbf{e}. \end{aligned}$$
- 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). It remains to prove Eq. (3.7). Note that $\Gamma^*_{\tau,2\ell,0}(\mathbf{x},\cdot)$ behaves exactly like $\Gamma^*_{\tau,2\ell,d}(\mathbf{x},\cdot)$ as long as one never makes a query $\mathbf{v}$ where $\mathbf{wt}(\mathbf{v} \wedge \overline{\mathbf{v}}^*) < d$. Since $\mathbf{v}^* \stackrel{\$}{\leftarrow} \{\mathbf{y} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{y}) = \ell\}$, for any $\mathbf{v}$, the probability that $\mathbf{wt}(\mathbf{v} \wedge \overline{\mathbf{v}}^*) < d$ is (by definition) $\alpha'_{\ell,d}$ as defined in Eq. (3.2). Using the union bound, we can upper bound the probability that $\mathbf{wt}(\mathbf{v} \wedge \overline{\mathbf{v}}^*) < d$ for any of the Q different $\mathbf{v}$'s chosen by the adversary as $Q \cdot \alpha'_{\ell,d}$. □

3.3 Avoid Checking

One disadvantage of the protocol in Figure 5, compared to HB style protocols, is the necessity to check whether the messages exchanged have the right from: 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. Eliminating such verification procedures can be particularly useful if for example the prover is an RFID chip where even the simple verification that a vector has large weight is expensive. We note that it is possible to eliminate these checks by blinding the exchanged messages $\mathbf{v}$ and $\mathbf{z}$ using random

$\mathcal{P}_{\tau,n}(\mathbf{s} \in \mathbb{Z}_2^{2\ell}, \mathbf{b^v} \in \mathbb{Z}_2^{2\ell}, \mathbf{b^z} \in \mathbb{Z}_2^n)$		$\mathcal{V}_{\tau',n}(\mathbf{s}, \mathbf{b^v}, \mathbf{b^z})$
	$\xleftarrow{\mathbf{v}}$	$\mathbf{v} \xleftarrow{\$} \mathbb{Z}_2^{2\ell}$
$\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{2\ell \times n}$; $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$		
$\mathbf{z} := \mathbf{R}^\mathsf{T} \cdot (\mathbf{s} \wedge (\mathbf{v} + \mathbf{b^v})) + \mathbf{b^z} + \mathbf{e}$	$\xrightarrow{(\mathbf{R},\mathbf{z})}$	if $\mathbf{wt}\big((\mathbf{R}^\mathsf{T} \cdot (\mathbf{s} \wedge (\mathbf{v} + \mathbf{b^v}))) + \mathbf{b^z}\big) > n \cdot \tau'$ reject otherwise accept

Figure 6: By blinding the values $\mathbf{v}, \mathbf{z}$ with secret random vectors $\mathbf{b^v}, \mathbf{b^z}$ we can avoid checking whether $\mathbf{wt}(\mathbf{v}) = \ell$ and $\mathsf{rank}(\mathbf{R}) = n$ as in the protocol from Figure 5.

vectors $\mathbf{b^v} \in \mathbb{Z}_2^{2\ell}$ and $\mathbf{b^z} \in \mathbb{Z}_2^n$ respectively, as shown in Figure 6. The security and completeness of this protocol is basically the same as for the protocol in Figure 6. The security proof is also very similar and is therefore omitted.

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 2-round authentication protocol from Section 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 $\Theta(\sqrt{\varepsilon} \cdot Q)$. However, the construction has the disadvantage that one needs to fix the hardness of the LPN problem at the time of the construction, c.f. Remark 4.3. Our second construction has no such issues and achieves better security $\Theta(\varepsilon \cdot Q)$. 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 2-round authentication protocol from Section 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 $\mathbf{m}$ to be authenticated as $\mathbf{v} = \mathsf{C}(h(\mathbf{m}, \mathbf{b}))$, where h is a pairwise independent hash function, $\mathbf{b} \in \mathbb{Z}_2^\nu$ is some fresh randomness and C is some encoding scheme. The code 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 $\mathbf{z} := \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow \mathbf{v}} + \mathbf{e}$, where $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$. We note that using $(\mathbf{R}, \mathbf{z})$ as an authentication tag would not be secure, and we need to blind these values. This is done by applying an (almost) pairwise independent permutation (PIP) π — which is part of the secret key — to $(\mathbf{R}, \mathbf{z}, \mathbf{b}) \in \mathbb{Z}_2^{\ell \times n + n + \nu}$.

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

- Public parameters. MAC_1 has the following public parameters.[7]

[7] The code 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 $\mathsf{C}(\mathbf{y}) = (\mathbf{c} \in \mathbb{Z}_2^\ell, \mathbf{c}' \in \mathbb{Z}_2^\ell)$ where $\mathbf{c} = \mathbf{C}^\mathsf{T} \cdot \mathbf{y}$ and $\mathbf{c}' = \bar{\mathbf{c}}$. A random code $\mathbf{C}$ has high distance with high probability and $\mathsf{C}(\mathbf{y}) = (\mathbf{c}, \mathbf{c}')$ has weight exactly ℓ.

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

- Key generation. Algorithm $\mathcal{K}(1^\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_2^{2\ell}$, an (almost) pairwise independent hash function $h : \mathcal{M} \times \mathbb{Z}_2^\nu \to \mathbb{Z}_2^\mu$ and a pairwise independent permutation π over $\mathbb{Z}_2^{\ell\times n+n+\nu}$. It returns $K = (\mathbf{s}, h, \pi)$ as the secret key.
- Tagging. Given secret key $K = (\mathbf{s}, h, \pi)$ and message $\mathbf{m} \in \mathcal{M}$, algorithm $\mathcal{T}$ proceeds as follows.
  1. $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell\times n}$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^\nu$, $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$
  2. $\mathbf{v} := \mathsf{C}(h(\mathbf{m}, \mathbf{b})) \in \mathbb{Z}_2^{2\ell}$
  3. Return $\phi := \pi(\mathbf{R}, \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow\mathbf{v}} + \mathbf{e}, \mathbf{b})$
- Verification. On input a secret-key $K = (\mathbf{s}, h, \pi)$, message $\mathbf{m} \in \mathcal{M}$ and tag ϕ, algorithm $\mathcal{V}$ proceeds as follows.
  1. Parse $\pi^{-1}(\phi)$ as $(\mathbf{R} \in \mathbb{Z}_2^{\ell\times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^\nu)$. If $\mathsf{rank}(\mathbf{R}) \neq n$, then return reject
  2. $\mathbf{v} := \mathsf{C}(h(\mathbf{m}, \mathbf{b}))$
  3. If $\mathbf{wt}(\mathbf{z} + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow\mathbf{v}}) > n \cdot \tau'$ return reject, otherwise return accept

Theorem 4.1. *For $\mu = \nu \in \mathbb{N}$, a constant $\gamma > 0$ and $d := \ell/(2+\gamma)$, if the $\mathsf{SLPN}^*_{\tau,2\ell,d}$ problem is (t, nQ, ε)-hard then MAC_1 is (t', Q, ε')-secure against uf-cma adversaries, where*

$$t' \approx t, \qquad \varepsilon = \min\left\{\varepsilon'/2 - \frac{Q^2}{2^{\mu-2}}\ ,\ \frac{\varepsilon'}{2^{\mu+1}} - 2^{-\Theta(n)}\right\}\ .$$

MAC_1 has completeness error $2^{-c_\tau \cdot n}$ where $c_\tau > 0$ depends only on τ.

Corollary 4.2. *Choosing μ s.t. $2^\mu = \frac{Q^2 \cdot 2^4}{\varepsilon'}$ in the above theorem, we get $\varepsilon = \min\{\varepsilon'/4, (\varepsilon')^2/(2^5 Q^2) - 2^{-\Theta(n)}\}$. The 2nd term is the minimum here, and solving for ε' gives*

$$\varepsilon' := \sqrt{32} \cdot Q \cdot \sqrt{\varepsilon + 2^{-\Theta(n)}}. \tag{4.1}$$

Remark 4.3 (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 $2^\mu \approx Q^2 \cdot 2^4/\varepsilon'$ 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 $\mathsf{SLPN}^*_{\tau,2\ell,d}$). 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 4.1. Every query $(\mathbf{m}, \phi)$ to $\mathcal{V}$ and query $\mathbf{m}$ 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 $(\mathbf{m}, \phi)$ is "fresh" if the $\mathbf{v}$ contained in $(\mathbf{m}, \phi)$ 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 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 $(\mathbf{m}, \phi)$ 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 $(\mathbf{m}, \phi)$ that contains a non-fresh $\mathbf{v}$ (even without being able to tell if $(\mathbf{m}, \phi)$ is valid or not).

If the 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 $\mathbf{v} \in \mathbb{Z}_2^{\mu}$ contained in the first forgery and cannot choose it as in the protocol. This is the reason why the reduction looses a factor 2^{μ}.

Proof of Theorem 4.1. 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 Figure 5 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 $\mathbf{Exp}^{\mathrm{uf-cma}}_{\mathsf{MAC}_1,\mathcal{A},\lambda}$ 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 $Q = Q_{\mathsf{tag}} + Q_{\mathsf{vrfy}}$. 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 $(\mathbf{m}, \phi)$ where ϕ was received as the output from the tag oracle on input $\mathbf{m}$. 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 $(\mathbf{m}, \phi)$ and tag query $\mathbf{m}$ defines 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 $(\mathbf{m}, \phi)$ where: (i) ϕ was not received as output on a tag query $\mathbf{m}$, and (ii) $\mathcal{V}(K, \mathbf{m}, \phi) = \mathsf{accept}$. We say that such a forgery $(\mathbf{m}, \phi)$ is "fresh" if the $\mathbf{v}$ defined by $(\mathbf{m}, \phi)$ is different from all $\mathbf{v}$'s defined by all the previous verification and tag queries. Let E_{fresh} denote the event that $\mathcal{A}$ finds a fresh forgery. As $\mathcal{A}$ finds a forgery with probability ε' and every forgery must be either fresh or not, we have that:

$$\Pr[E_{\mathsf{fresh}}] + \Pr[\neg E_{\mathsf{fresh}}] = \varepsilon'.$$

We will consider the two cases where $\Pr[E_{\mathsf{fresh}}] > \varepsilon'/2$ and $\Pr[E_{\mathsf{fresh}}] \leq \varepsilon'/2$ separately.

The case $\Pr[E_{\mathsf{fresh}}] \leq \varepsilon'/2$. Given $\mathcal{A}$, we will construct an adversary $\mathcal{B}_1^{\mathcal{O}}$ who can distinguish $\mathcal{O} = \Gamma^*_{\tau,2\ell,d}(\mathbf{s}, \cdot)$ from $\mathcal{O} = U_{2\ell+1}(\cdot)$ (as in Definition 2.7) with advantage[8]

$$\varepsilon'/2 - \frac{Q^2}{2^{\mu-2}}. \tag{4.2}$$

$\mathcal{B}_1^{\mathcal{O}}$ 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 $(\mathbf{s}, h, \pi)$) answering its queries as follows:

- **Tag queries.** If $\mathcal{A}$ makes a tag query $\mathbf{m}$, then $\mathcal{B}_1^{\mathcal{O}}$ does the following:
    1. Sample $\mathbf{b} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\nu}$ and compute $\mathbf{v} := \mathsf{C}(h(\mathbf{m}, \mathbf{b}))$.

[8] In this case where $\Pr[E_{\mathsf{fresh}}] \leq \varepsilon'/2$, we can even distinguish a $\mathsf{SLPN}^*_{\tau,2\ell,\ell}$ oracle from $U_{2\ell+1}(\cdot)$.

2. Query the oracle $\mathcal{O}$ for n times on input $\mathbf{v}$: for $i = 1, \ldots, n$ let $(\mathbf{R}[i], \mathbf{z}[i]) \stackrel{\$}{\leftarrow} \mathcal{O}(\mathbf{v})$.
3. Return $\phi := \pi(\mathbf{R}, \mathbf{z}, \mathbf{b})$ where $\mathbf{R} = [\mathbf{R}[1], \ldots, \mathbf{R}[n]]$ and $\mathbf{z} = [\mathbf{z}[1], \ldots, \mathbf{z}[n]]$ to $\mathcal{A}$.

- **Verification queries.** If $\mathcal{A}$ makes a verification query $(\mathbf{m}, \phi)$, $\mathcal{B}_1^{\mathcal{O}}$ 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 4.4. *If $\mathcal{O} = \Gamma^*_{\tau,2\ell,d}(\mathbf{s}, \cdot)$, 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 $(\mathbf{s}, h, \pi)$). The answers to the verification queries (which are always reject) are correct as long as $\mathcal{A}$ does not query a valid forgery. From our assumption, the probability that $\mathcal{A}$ finds a valid forgery that is *not* fresh is $> \varepsilon'/2$, which is thus a lower bound on the probability that $\mathcal{B}_1^{\mathcal{O}}$ outputs 1. □

Claim 4.5. *If $\mathcal{O} = U_{2\ell+1}(\cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $< Q^2/2^{\mu-2}$.*

Proof of Claim. The answers that $\mathcal{A}$ obtains on a tag query $\mathbf{m}$ 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 $\mathbf{v}_i = \mathsf{C}(h(\mathbf{m}_i, \mathbf{b}_i))$ is defined by the i-th tag or verification query) iff $h(\mathbf{m}_i, \mathbf{b}_i) = h(\mathbf{m}_j, \mathbf{b}_j)$.

$\mathcal{A}$ makes a total of Q queries. Assume that up to the $(i-1)$th query, all the $\mathbf{v}$'s were distinct. If the ith query is a tag query, a fresh $\mathbf{b}_i$ is sampled which will be distinct from all previous $\mathbf{b}_j$ (for any $j < i$) with probability $1 - (i-1)/2^{\nu}$. Assuming this is the case, the probability that $h(\mathbf{m}_i, \mathbf{b}_i) = h(\mathbf{m}_j, \mathbf{b}_j)$ for any $j < i$ can be upper bounded by $i/2^{\mu}$ (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 h).

If the ith query is a verification query $(\mathbf{m}_i, \phi_i)$, then using the fact that π is a pairwise independent permutation (and $\mathcal{A}$ has no information about it) we can show that the probability that ϕ_i contains a $\mathbf{b}_i$ which is equal to some $\mathbf{b}_j$ (s.t. $\phi_j \neq \phi_i$) is $\leq i/2^{\nu+1}$. If this is the case then $(\mathbf{m}_i, \mathbf{b}_i) \neq (\mathbf{m}_j, \mathbf{b}_j)$ for all $j < i$ with overwhelming probability.[9] As in the previous case, we can then upper bound the probability that $h(\mathbf{m}_i, \mathbf{b}_i) = h(\mathbf{m}_j, \mathbf{b}_j)$ for any $j < i$ by $i/2^{\mu}$.

Using the union bound over all $i, 1 \leq i \leq Q$ we get the bound $Q^2/2^{\nu-2} = Q^2/2^{\mu-2}$ (recall that $\mu = \nu$) as claimed. □

The case $\Pr[E_{\mathsf{fresh}}] > \varepsilon'/2$. 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 $(\mathbf{m}, \phi)$ 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 $\mathcal{O} = \Gamma^*_{\tau,2\ell,d}(\mathbf{s}, \cdot)$ from $\mathcal{O} = U_{2\ell+1}(\cdot)$ (as in Definition 2.7) with advantage

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

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 Figure 5 is secure against active attacks). The queries to the prover in the first phase of an active attack directly correspond to tag

[9]Note that for $j < i$ where $\phi_i = \phi_j$ we must have that $\mathbf{m}_i \neq \mathbf{m}_j$ since we assume that $\mathcal{A}$ does not repeat queries and does not ask verification queries $(\mathbf{m}, \phi)$ if ϕ was the output of a tag query $\mathbf{m}$.

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 $\varepsilon'/2^{\mu+1}$ as stated in Eq. (4.3). We do loose an additional additive term $Q_{\mathsf{tag}} \cdot \alpha'_{\ell,d}$ 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 $Q_{\mathsf{vrfy}} \cdot \alpha''_{\tau',n}$ 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 $\mathbf{y}^* \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\mu$, $\mathbf{s}^* \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{2\ell}$. Let $\mathbf{v}^* := \mathsf{C}(\mathbf{y}^*)$. 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 4.7).

- **Tag queries.** The answer ϕ to a tag query $\mathbf{m} \in \mathcal{M}$ is computed by $\mathcal{B}_2^{\mathcal{O}}$ as follows:
  1. Sample $\mathbf{b} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^\nu$ and compute $\mathbf{v} := \mathsf{C}(h(\mathbf{m}, \mathbf{b}))$. If $\mathbf{v} = \mathbf{v}^*$, output 0 and stop. Let $\mathbf{u} := \mathbf{v} \wedge \overline{\mathbf{v}}^*$ and $\mathbf{u}^* := \mathbf{v} \wedge \mathbf{v}^*$.
  2. For $i = 1, \ldots, n$, let $(\mathbf{R}'[i], \mathbf{z}'[i]) \stackrel{\$}{\leftarrow} \mathcal{O}(\mathbf{u})$, $\mathbf{R}''[i] \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{2\ell}$ and $\mathbf{z}''[i] := \langle \mathbf{R}''[i], \mathbf{s}^* \wedge \mathbf{u}^* \rangle$. Let $\mathbf{R} = [\mathbf{R}[1], \mathbf{R}[2], \ldots, \mathbf{R}[n]]$ and $\mathbf{z} = [\mathbf{z}[1], \ldots, \mathbf{z}[n]]$ 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 $(\phi, \mathbf{m})$, then $\mathcal{B}_2^{\mathcal{O}}$ always answers reject, but also makes the following check:
  1. Parse $\mathbf{y} := \pi^{-1}(\phi)$ as $[\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^\nu]$ and compute $\mathbf{v} := \mathsf{C}(h(\mathbf{m}, \mathbf{b}))$.
  2. If $\mathbf{v} \neq \mathbf{v}^*$, processing this query is over, otherwise go to the next step.
  3. If $\mathsf{rank}(\mathbf{R}) = n$ and $\mathbf{wt}(\mathbf{R}^\mathsf{T} \cdot \mathbf{s}^*_{\downarrow \mathbf{v}^*} + \mathbf{z}) \leq n \cdot \tau'$ (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 4.6. *If $\mathcal{O} = U_{2\ell+1}(\cdot)$, then $\mathcal{B}_2^{\mathcal{O}}$ outputs 1 with probability $\leq Q_{\mathsf{vrfy}} \cdot \alpha''_{\tau',n}$.*

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. □

Claim 4.7. *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. $\mathcal{B}_2^{\Gamma^*_{\tau,2\ell,d}(\mathbf{s},\cdot)}$ perfectly simulates access to $\mathcal{T}(K,\cdot), \mathcal{V}(K,\cdot,\cdot)$ oracles with key $K = (\mathbf{s}', h, \pi)$ where $\mathbf{s}' := (\mathbf{s}^* \wedge \mathbf{v}^*) + (\mathbf{s} \wedge \overline{\mathbf{v}}^*)$ and h, π are sampled by $\mathcal{B}_2^{\mathcal{O}}$. By assumption, in this case, $\mathcal{A}$ outputs a valid fresh forgery with probability $\varepsilon'/2$. Conditioned on this, with probability $2^{-\mu}$, this fresh $\mathbf{v}$ will be $\mathbf{v}^*$ and therefore $\mathcal{B}_2^{\mathcal{O}}$ will output 1. □

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.

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

□

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 [3], that we needed to adapt to the case of LPN.

Construction. The message authentication code $\mathsf{MAC}_2 = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ 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 Section 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 a pairwise independent hash function $h : \mathcal{M} \times \mathbb{Z}_2^{\nu} \to \mathbb{Z}_2^{\mu} \setminus \{0\}$, as well as a pairwise independent permutation π over $\mathbb{Z}_2^{\ell \times n + n + \nu}$. 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 $\mathbf{m} \in \mathcal{M}$, algorithm $\mathcal{T}$ proceeds as follows.
  1. $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^{\nu}$, $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$
  2. $\mathbf{v} := h(\mathbf{m}, \mathbf{b})$
  3. $\mathbf{s}(\mathbf{v}) := \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$
  4. Return $\phi := \pi(\mathbf{R}, \mathbf{s}_0' + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{v}) + \mathbf{e}, \mathbf{b})$
- Verification. On input a secret-key $K = (\mathbf{S}, \mathbf{s}_0, \mathbf{s}_0', h, \pi)$, message $\mathbf{m} \in \mathcal{M}$ and tag ϕ, algorithm $\mathcal{V}$ proceeds as follows.
  1. Parse $\pi^{-1}(\phi)$ as $(\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}, \mathbf{z} \in \mathbb{Z}_2^n, \mathbf{b} \in \mathbb{Z}_2^{\nu})$. If $\mathsf{rank}(\mathbf{R}) \neq n$, then return reject
  2. $\mathbf{v} := h(\mathbf{m}, \mathbf{b})$
  3. $\mathbf{s}(\mathbf{v}) := \mathbf{s}_0 + \mathbf{S} \cdot \mathbf{v}$
  4. If $\mathbf{wt}(\mathbf{z} + \mathbf{s}_0' + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{v})) > n \cdot \tau'$ return reject, otherwise return accept

Theorem 4.8. *Let $\nu = \mu$. If the $\mathsf{SLPN}_{\tau,\ell,\ell}$ problem is (t, nQ, ε)-hard, then MAC_2 is (t', Q, ε')-secure against uf-cma adversaries, where*

$$t' \approx t \qquad \varepsilon = \min\left\{\varepsilon'/2 - \frac{Q^2}{2^{\mu-2}}, \frac{\varepsilon'}{8\mu Q} - 2^{-\Theta(n)}\right\}.$$

MAC_2 has completeness error $2^{-c_\tau \cdot n}$ where c_τ only depends on τ.

We now give intuition for the proof of Theorem 4.8. Similar to the proof of Theorem 4.1, we distinguish fresh and non-fresh forgeries. Here the new and interesting case is when the adversary makes a fresh forgery. 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 defined an indistinguishable distribution, assuming that the LPN assumption holds. More in detail, consider the two experiments defined in Figure 7. In the "real experiment", the answers from the $\textsc{Eval}(\mathbf{v})$ oracle have the same distribution as the values $(\mathbf{R}, \mathbf{z})$ from a tag on message $\mathbf{m}$ such that $h(\mathbf{m}, \mathbf{b}) = \mathbf{v}$; in the "random experiment", the answers

$\underline{\mathbf{Exp}^{\mathsf{real}}_{\ell,\mu,n,\tau}(\mathcal{B}),\ \mathbf{Exp}^{\mathsf{rand}}_{\ell,\mu,n,\tau}(\mathcal{B})}$

$\mathcal{Q}_{\mathbf{v}} := \emptyset$

$\mathbf{S} \xleftarrow{\$} \mathbb{Z}_2^{\ell\times\mu}; \mathbf{s}_0 \xleftarrow{\$} \mathbb{Z}_2^{\ell}; \mathbf{s}_0' \xleftarrow{\$} \mathbb{Z}_2^{n}$

$\{0,1\} \ni d \leftarrow \mathcal{B}^{\textsc{Eval}(\cdot),\textsc{Chal}(\cdot,\cdot)}(1^\lambda)$

Return $d \wedge (\mathbf{v}^* \notin \mathcal{Q}_{\mathbf{v}})$

$\underline{\text{Oracle } \textsc{Chal}(\mathbf{R}^*, \mathbf{v}^*)}$ // one query

$\mathbf{s}(\mathbf{v}^*) = \mathbf{s}_0 + \mathbf{S}\cdot\mathbf{v}^*$

Return $\mathbf{z}^* = \mathbf{s}_0' + \mathbf{R}^{*\mathsf{T}}\cdot\mathbf{s}(\mathbf{v}^*)$

$\underline{\text{Oracle } \textsc{Eval}(\mathbf{v})}$

$\mathcal{Q}_{\mathbf{v}} := \mathcal{Q}_{\mathbf{v}} \cup \{\mathbf{v}\}$

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

$\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell\times n}; \mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$

$\mathbf{z} = \mathbf{s}_0' + \mathbf{R}^{\mathsf{T}}\cdot\mathbf{s}(\mathbf{v}) + \mathbf{e}$; $\boxed{\mathbf{z} \xleftarrow{\$} \mathbb{Z}_2^n}$

Return $(\mathbf{R}, \mathbf{z})$

Figure 7: Experiments $\mathbf{Exp}^{\mathsf{real}}_{\ell,\mu,n,\tau}(\mathcal{B})$ and $\boxed{\mathbf{Exp}^{\mathsf{rand}}_{\ell,\mu,n,\tau}(\mathcal{B})}$. The boxed statement redefining $\mathbf{z}$ is only executed in $\mathbf{Exp}^{\mathsf{rand}}_{\ell,\mu,n,\tau}$.

from the $\textsc{Eval}(\mathbf{v})$ oracle are uniform. Oracle $\textsc{Chal}(\mathbf{R}^*, \mathbf{v}^*)$, which can be queried at most once, essentially corresponds to the output of a verification query on a *fresh* forgery $(\mathbf{m}^*, \phi^*)$, such that $h(\mathbf{m}^*, \mathbf{b}^*) = \mathbf{v}^*$. The lemma below states that it is hard to distinguish the two cases. Its proof uses a hybrid technique from [10, 3].

Lemma 4.9. *Let $\ell, \mu, n, \tau \in \mathbb{N}$. Assume that the $\mathsf{LPN}_{\tau,\ell}$ problem is (t, nQ, ε)-hard. Then, for all adversaries $\mathcal{B}$ running in time $t' \approx t$, and asking Q queries to the $\textsc{Eval}(\cdot)$ oracle, we have that*

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

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

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

This is obtained by letting $\mathbf{s}_0 = \sum_{j=1}^{\mu} \mathbf{s}(j,0)$, and $\mathbf{S}[j] = \mathbf{s}_{j,1} - \mathbf{s}_{j,0}$.

Let $\mathbf{G}_0$ be identical to the "real experiment" $\mathbf{Exp}^{\mathsf{real}}_{\ell,\mu,n,\tau}(\mathcal{B})$, with the difference that the vectors $\mathbf{s}_{i,j}$ (as defined above) are used, instead of $(\mathbf{S}, \mathbf{s}_0)$, to define $\mathbf{s}(\mathbf{v})$. We prove the lemma by considering a sequence of intermediate games, starting with game $\mathbf{G}_0$. The games are shown in Figure 8. Note that in Game $\mathbf{G}_{1,0}$ the value $\mathbf{s}_0'$ is computed as $\mathsf{RF}_0(\perp)$, where $\perp$ is the empty string, that always outputs the same vector $\mathbf{s}_0' \xleftarrow{\$} \mathbb{Z}_2^\ell$. Therefore, we have

Claim 4.10. $\Pr[\mathbf{G}_0 = 1] = \Pr[\mathbf{G}_{1,0} = 1]$.

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

Claim 4.11. *There exists a distinguisher $\mathcal{D}$, with running time similar to that of $\mathcal{B}$, such that*

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

Proof. Let Q be the total number of queries that $\mathcal{B}$ asks to the $\textsc{Eval}(\cdot)$ 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 $U_{\ell+1}$.

Distinguisher $\mathcal{D}$ starts by sampling a bit $b \xleftarrow{\$} \{0,1\}$ as its guess for $\mathbf{v}^*[i]$ and defines the random function $\mathsf{RF}_{i+1}(\cdot)$ recursively as

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

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

1. At setup $\mathcal{D}$ does the following:
   - Sample $b \xleftarrow{\$} \{0,1\}$ and set $\mathcal{Q}_{\mathbf{v}} := \emptyset$.
   - Choose all the vectors $\mathbf{s}_{j,k} \xleftarrow{\$} \mathbb{Z}_2^\ell$ at random (for all $j \in \{1, \dots, \mu\}$ and $k \in \{0,1\}$), but $\mathbf{s}_{i+1,1-b}$ 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 $\alpha_i : \{0,1\}^i \to [Q]$ be an injective function.
2. Upon input a query $\mathbf{v}$ to oracle $\textsc{Eval}(\cdot)$, distinguisher $\mathcal{D}$ does the following:
   - Update $\mathcal{Q}_{\mathbf{v}} := \mathcal{Q}_{\mathbf{v}} \cup \{\mathbf{v}\}$.
   - If $\mathbf{v}[i+1] = b$, let $\mathsf{RF}_{i+1}(\mathbf{v}[1 \dots i+1]) = \mathsf{RF}_i(\mathbf{v}[1 \dots i])$. Sample $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$, $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$, compute $\mathbf{z} = \mathsf{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^\mathsf{T} \cdot (\sum_{j=1}^\mu \mathbf{s}_{j,\mathbf{v}[j]}) + \mathbf{e}$ and return $(\mathbf{R}, \mathbf{z})$.
   - Else, in case $\mathbf{v}[i+1] = 1-b$, let $(\mathbf{R}, \mathbf{z}') := (\mathbf{R}_j, \mathbf{z}'_j)$ for $j = \alpha_i(\mathbf{v}[1 \dots i])$. Define

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

     and return $(\mathbf{R}, \mathbf{z})$.
3. Upon input query $(\mathbf{R}^*, \mathbf{v}^*)$ to oracle $\textsc{Chal}(\cdot)$, distinguisher $\mathcal{D}$ does the following:
   - Define $\mathbf{s}(\mathbf{v}^*) = \sum_{j=1}^\mu \mathbf{s}_{j,\mathbf{v}[j]}$.
   - Return $\mathbf{z}^* = \mathbf{R}^{*\mathsf{T}} \cdot \mathbf{s}(\mathbf{v}^*) + \mathsf{RF}_i(\mathbf{v}^*[1 \dots i])$.
4. Upon input the decision bit d from $\mathcal{B}$, distinguisher $\mathcal{D}$ returns $d \wedge (\mathbf{v}^* \not\in \mathcal{Q}_{\mathbf{v}})$.

Suppose that $\mathcal{D}$ correctly guessed $\mathbf{v}^*[i]$, which happens with probability $1/2$. Note that in this case $\mathcal{D}$ simulates perfectly the answer of the $\textsc{Chal}(\cdot)$ oracle (as it knows $\mathbf{s}_{i+1,b}$). It remains to analyze the distribution of oracle $\textsc{Eval}(\cdot)$. 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 $\mathsf{RF}_{i+1}(\mathbf{v}[1 \dots i+1]) = \mathsf{RF}_i(\mathbf{v}[1 \dots i])$). 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 $\mathbf{z}' = \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{i+1,1-b} + \mathbf{e}$ and thus the answer

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

is distributed like in game $\mathbf{G}_i$. In the second case, we have $\mathbf{z}' = \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{i+1,1-b} + \mathbf{e} + \mathbf{u}$ (for a uniform $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_2^\ell$). Thus, the answer $\mathbf{z}$ computed by $\mathcal{D}$ is distributed like in $\mathbf{G}_{i+1}$ with random

$\mathbf{G}_0$, $\mathbf{G}_{1,i}$, $\mathbf{G}_2$

$\mathcal{Q}_{\mathbf{v}} := \emptyset$
$\mathbf{s}_{j,0}, \mathbf{s}_{j,1} \xleftarrow{\$} \mathbb{Z}_2^{\ell}, \forall j \in [\mu]$
$\mathbf{s}'_0 \xleftarrow{\$} \mathbb{Z}_2^{n}$; $\boxed{\mathbf{s}'_0 := \bot}$
$\{0,1\} \ni d \leftarrow \mathcal{B}^{\text{EVAL}(\cdot),\text{CHAL}(\cdot,\cdot)}(1^{\lambda})$
Return $d \wedge (\mathbf{v}^* \notin \mathcal{Q}_{\mathbf{v}})$

Oracle $\text{CHAL}(\mathbf{R}^*, \mathbf{v}^*)$ // one query

$\boxed{\mathbf{s}'_0 = \mathsf{RF}_i(\mathbf{v}^*[1 \dots i])}$
$\mathbf{s}(\mathbf{v}^*) = \sum_{j=1}^{\mu} \mathbf{s}_{j,\mathbf{v}^*[j]}$
$\mathbf{z}^* = \mathbf{s}'_0 + \mathbf{R}^{*\mathsf{T}} \cdot \mathbf{s}(\mathbf{v}^*)$; $\colorbox{lightblue}{$\mathbf{z}^* \xleftarrow{\$} \mathbb{Z}_2^{\ell}$}$
Return $\mathbf{z}^*$

Oracle $\text{EVAL}(\mathbf{v})$

$\mathcal{Q}_{\mathbf{v}} := \mathcal{Q}_{\mathbf{v}} \cup \{\mathbf{v}\}$; $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell \times n}$; $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_{\tau}^{n}$
$\mathbf{z} = \mathbf{s}'_0 + \mathbf{R}^{\mathsf{T}} \cdot (\sum_{j=1}^{\mu} \mathbf{s}_{j,\mathbf{v}[j]}) + \mathbf{e}$
$\boxed{\mathbf{z} = \mathsf{RF}_i(\mathbf{v}[1 \dots i]) + \mathbf{R}^{\mathsf{T}} \cdot (\sum_{j=1}^{\mu} \mathbf{s}_{j,\mathbf{v}[j]}) + \mathbf{e}}$
$\colorbox{lightblue}{$\mathbf{z} = \mathsf{RF}_{\mu}(\mathbf{v})$}$
Return $(\mathbf{R}, \mathbf{z})$

Figure 8: Hybrid experiments $\mathbf{G}_0$, $\boxed{\mathbf{G}_{1,i}}$ and $\colorbox{lightblue}{$\mathbf{G}_2$}$ in the proof of Lemma 4.9. Here $\mathsf{RF}_i : \{0,1\}^i \leftarrow \mathbb{Z}_2^{\ell}$ 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 $\mathbf{v} \in \mathbb{Z}_2^{\mu}$

function $\mathsf{RF}'_i(\mathbf{v}[1 \dots i]) = \mathbf{u}$. Note that RF'_i is well-defined, i.e., the value $\mathsf{RF}'_i(\mathbf{v}[1, \cdots i])$ 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 α_i is an injection). The claim follows. □

Claim 4.12. $\Pr[\mathbf{G}_{1,\mu} = 1] = \Pr[\mathbf{G}_2 = 1]$.

Proof. The claim follows from the fact that in $\mathbf{G}_{1,\mu}$ all outputs computed via $\text{EVAL}(\cdot)$ are masked by $\mathsf{RF}_{\mu}(\mathbf{v})$ 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 4.11 and is therefore omitted.

Claim 4.13. *There exists a distinguisher $\mathcal{D}$, with running time similar to that of $\mathcal{B}$, such that*

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

The statement of Lemma 4.9 now follows by putting together Claim 4.10—4.13.

□

We now turn to the proof of Theorem 4.8.

Proof of Theorem 4.8. The proof of the completeness error is similar to the schemes before and is omitted. As for security, let $\mathcal{A}$ be an adversary that successfully forges in the uf-cma experiment with probability ε'. We make the same conventions and the definition of freshness as in the proof of Theorem 4.1 and split the forging probability as $\Pr[E_{\mathsf{fresh}}] + \Pr[\neg E_{\mathsf{fresh}}] = \varepsilon'$.

The case $\Pr[E_{\mathsf{fresh}}] \leq \varepsilon'/2$. We now give the description of $\mathcal{B}_1^{\mathcal{O}}$ attacking the $\mathsf{SLPN}_{\tau,\ell,\ell}$ 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

$$\varepsilon'/2 - \frac{Q^2}{2^{\mu-2}}. \tag{4.5}$$

Adversary $\mathcal{B}_1^{\mathcal{O}}$ samples $\pi, h, \mathbf{s}_0, \mathbf{s}'_0$ (but not $\mathbf{S}$) as defined by $\mathcal{K}$ and $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_2^{\ell\times\mu}$. Next, it implicitly defines $\mathbf{S}$ as $\mathbf{S}[i] = \mathbf{I}\cdot\mathbf{s}+\mathbf{B}[i]$, where $\mathbf{I}$ is the identity matrix, and $\mathbf{s}$ 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 $\mathbf{v} \in \mathbb{Z}_2^{\mu} \setminus \{0\}$,

$$\mathbf{s}(\mathbf{v}) = \mathbf{s}_0 + \mathbf{S}\cdot\mathbf{v} = \mathbf{A}(\mathbf{v})\cdot\mathbf{s} + \mathbf{b}(\mathbf{v}),\ \text{where } \mathbf{A}(\mathbf{v}) = \mathbf{wt}(\mathbf{v})\cdot\mathbf{I} \text{ and } \mathbf{b}(\mathbf{v}) = \mathbf{s}_0 + \mathbf{B}\cdot\mathbf{v}. \tag{4.6}$$

Note that $\mathbf{b}(\mathbf{v})$ and $\mathbf{A}(\mathbf{v})$ are known to $\mathcal{B}_1$ and that, by construction, $\mathbf{A}(\mathbf{v})$ is always an invertible matrix. 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 $\mathbf{m} \in \mathcal{M}$, then $\mathcal{B}_1^{\mathcal{O}}$ does the following:
  1. Samples $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^{\nu}$ and compute $\mathbf{v} := h(\mathbf{m}, \mathbf{b})$.
  2. Query the oracle $\mathcal{O}$ on $(\mathbf{A}(\mathbf{v}), \mathbf{b}(\mathbf{v}))$ n times to obtain $(\mathbf{R}, \mathbf{z}')$: for $i = 1, \ldots, n$ let $(\mathbf{R}[i], \mathbf{z}'[i]) \xleftarrow{\$} \mathcal{O}(\mathbf{s}, \mathbf{A}(\mathbf{v}), \mathbf{b}(\mathbf{v}))$.
  3. Return $\phi := \pi(\mathbf{R}, \mathbf{s}'_0 + \mathbf{z}', \mathbf{b})$.
- **Verification queries.** If $\mathcal{A}$ makes a verification query $(\mathbf{m}, \phi)$, $\mathcal{B}_1^{\mathcal{O}}$ 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 $\mathbf{z}'[i] = \mathbf{R}[i]^{\mathsf{T}}(\mathbf{A}(\mathbf{v})\cdot\mathbf{s} + \mathbf{b}(\mathbf{v})) + \mathbf{e}[i] = \mathbf{R}[i]^{\mathsf{T}}\cdot\mathbf{s}(\mathbf{v}) + \mathbf{e}[i]$.

The following two claims are the analogues of Claims 4.4 and 4.5, respectively. Their proofs are essentially the same and are therefore omitted.

Claim 4.14. *If $\mathcal{O} = \Gamma_{\tau,\ell}(\mathbf{s},\cdot,\cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $\geq \varepsilon'/2$.*

Claim 4.15. *If $\mathcal{O} = U_{\ell+1}(\cdot,\cdot)$, then $\mathcal{B}_1^{\mathcal{O}}$ outputs 1 with probability $< \frac{Q^2}{2^{\mu-2}}$.*

The case $\Pr[E_{\mathsf{fresh}}] > \varepsilon'/2$. We will use games, denoting by $\mathbf{G}_i$ the output of the ith game. Game $\mathbf{G}_0$ runs the uf-cma security experiment $\mathbf{Exp}^{\text{uf}-\text{cma}}_{\mathsf{MAC}_2,\mathcal{A},\lambda}$ and defines the output as the event E_{fresh}. By definition we have $\Pr[\mathbf{G}_0 = 1] = \Pr[E_{\mathsf{fresh}}] \geq \varepsilon'/2$. Throughout the rest of the proof, if in the game $\mathcal{A}$ finds a forgery, and the first forgery is fresh, we'll 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 $(\mathbf{m}^*, \phi^*)$. 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. We assume that after $\mathbf{G}_0$ processes a verification query with respect to $\mathbf{v}^*$, the random variable corresponding to the outcome of the game is defined and the experiment stops.

Assume that $\mathcal{A}$ asks a total of $Q = Q_{\mathsf{tag}} + Q_{\mathsf{vrfy}}$ queries, where Q_{tag} (resp., Q_{vrfy}) stands for the total number of queries asked to the tag (resp., verification) oracle. Define E^j_{fresh} 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 $(\mathbf{m}, \phi)$ previously returned by the tag oracle. Since all the events E^j_{fresh} are disjoint, for $j \in \{1, \ldots, Q_{\mathsf{vrfy}}\}$ we have:

$$\Pr[\mathbf{G}_0 = 1] = \Pr[E_{\mathsf{fresh}}] = \Pr\left[\bigcup_{j=1}^{Q_{\mathsf{vrfy}}} E^j_{\mathsf{fresh}}\right] = \sum_{j=1}^{Q_{\mathsf{vrfy}}} \Pr\left[E^j_{\mathsf{fresh}}\right].$$

We now consider games $\mathbf{G}_1, \ldots, \mathbf{G}_{Q_{\mathsf{vrfy}}}$ where game $\mathbf{G}_j$ is identical to $\mathbf{G}_0$, but allows the adversary $\mathcal{A}$ to ask only j verification queries, and the answer to the first $j-1$ verification queries is always reject unless the input is a pair $(\mathbf{m}, \phi)$ already returned by the tag oracle (in which case we answer with accept). It is easy to see that $\Pr[\mathbf{G}_j = 1] \geq \Pr[E^j_{\mathsf{fresh}}] - (j-1)\alpha$, where the offset depending on the completeness error $\alpha = 2^{-\Theta(n)}$ of MAC_2 comes from the fact that in $\mathbf{G}_j$ we always return accept in case of a verification query for a pair $(\mathbf{m}, \phi)$ previously returned by the tag oracle. Plugging this expression in the previous equation, we obtain

$$\Pr[\mathbf{G}_0 = 1] \leq Q^2_{\mathsf{vrfy}}\alpha + \sum_{j=1}^{Q_{\mathsf{vrfy}}} \Pr[\mathbf{G}_j = 1].$$

In the remainder of the proof, we will upper bound $\Pr[\mathbf{G}_j = 1]$, for all $j \in \{1, \ldots, Q_{\mathsf{vrfy}}\}$. Fix a value of $j \in \{1, \ldots, Q_{\mathsf{vrfy}}\}$. As a first step, consider a modified version $\mathbf{G}'_j$ of Game $\mathbf{G}_j$ where the tag oracle internally uses uniform $(\mathbf{R}, \mathbf{z}) \in \mathbb{Z}_2^{\ell \times n} \times \mathbb{Z}_2^n$ to generate tag ϕ on message $\mathbf{m}$.

Claim 4.16. $\left|\Pr[\mathbf{G}_j = 1] - \Pr[\mathbf{G}'_j = 1]\right| \leq 4\mu\varepsilon.$

Proof. 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

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

contradicting Lemma 4.9. Adversary $\mathcal{B}$ works as follows.

1. At the beginning $\mathcal{B}$ samples h, π (but not $\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0$).
2. Upon input a query $\mathbf{m}$ to the tag oracle, $\mathcal{B}$ does the following:
   - Sample a random $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^\nu$ and compute $\mathbf{v} = h(\mathbf{m}, \mathbf{b})$.
   - Query $\mathbf{v}$ to oracle $\textsc{Eval}(\cdot)$, 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 $(\mathbf{m}, \phi)$ to the verification oracle, $\mathcal{B}$ does the following:
   - First check whether $(\mathbf{m}, \phi)$ is equal to one of the tags previously returned to $\mathcal{D}$; if this is the case answer with accept.
   - Otherwise, check whether $(\mathbf{m}, \phi)$ is the j-th verification query; if this is not the case, then answer with reject.
   - Else, $(\mathbf{m}, \phi)$ is the j-th verification query; call it $(\mathbf{m}^*, \phi^*)$. Let $(\mathbf{R}^*, \mathbf{z}^*, \mathbf{b}^*) = \pi^{-1}(\phi^*)$, compute $\mathbf{v}^* = h(\mathbf{m}^*, \mathbf{b}^*)$ and forward $(\mathbf{R}^*, \mathbf{v}^*)$ to oracle $\textsc{Chal}(\cdot)$ 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 $\textsc{Chal}(\cdot)$ oracle, so has the right distribution. The claim follows. □

Claim 4.17. $\Pr[\mathbf{G}'_j = 1] \leq \alpha''_{\tau',n} = 2^{-\Theta(n)}$.

Proof of Claim. If $\mathbf{R}^*$ does not have full rank then the experiment outputs 0 by definition. So from now we only consider the case where $\mathsf{rank}(\mathbf{R}^*) = n$. 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 $(\mathbf{S}, \mathbf{s}_0, \mathbf{s}'_0)$. Since $\mathbf{s}(\mathbf{v}^*)$ is uniformly random and $\mathbf{R}^*$ has full rank, the vector $\mathbf{x} := \mathbf{s}'_0 + \mathbf{R}^{*\mathsf{T}} \cdot \mathbf{s}(\mathbf{v}^*) + \mathbf{z}^*$ is uniformly random over $\mathbb{Z}_2^n$. Thus the probability that the second verification $\mathbf{wt}(\mathbf{z}^* + \mathbf{s}'_0 + \mathbf{R}^{*\mathsf{T}} \cdot \mathbf{s}(\mathbf{v}^*)) \leq n \cdot \tau'$ passes is $\Pr[\mathbf{wt}(\mathbf{x}) \leq n \cdot \tau'] = \alpha''_{\tau',n} = 2^{-\Theta(n)}$. □

Summing up, in the case $\Pr[E_{\mathsf{fresh}}] > \varepsilon'/2$ (putting together the terms in Claim 4.16—4.17), we can use $\mathcal{A}$ to break the LPN assumption with advantage $\frac{\varepsilon'}{4\mu Q} - 2^{-\Theta(n)}$. On the other hand in the case $\Pr[E_{\mathsf{fresh}}] \leq \varepsilon'/2$, we have an advantage as given in Eq. (4.5). Thus

$$\varepsilon = \min\left\{\varepsilon'/2 - \frac{Q^2}{2^{\mu-2}}, \frac{\varepsilon'}{4\mu Q} - 2^{-\Theta(n)}\right\},$$

as desired. □

Acknowledgements

Krzysztof would like to thank Vadim Lyubashevsky for many interesting discussions on LPN while being in Tel Aviv and Eyjafjallajökull for making this stay possible.

References

[1] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, May 2010.

[2] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[3] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. (Hierarchical) identity-based encryption from affine message authentication. In *CRYPTO*, pages ???–???, 2014.

[4] Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 278–291. Springer, August 1994.

[5] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. In *32nd ACM STOC*, pages 435–440. ACM Press, May 2000.

[6] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003.

[7] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, May 2004.

[8] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 499–517. Springer, May 2010.

[9] Julien Bringer, Hervé Chabanne, and Emmanuelle Dottax. HB^{++}: a lightweight authentication protocol secure against some attacks. In *SecPerU*, pages 28–33, 2006.

[10] Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure ibe and dual system groups. In *CRYPTO (2)*, pages 435–460, 2013.

[11] Ronald Cramer and Ivan Damgaard. On the amortized complexity of zero-knowledge protocols. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 177–191. Springer, August 2009.

[12] Yevgeniy Dodis, Eike Kiltz, Krzysztof Pietrzak, and Daniel Wichs. Message authentication, revisited. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 355–374. Springer, 2012.

[13] D.N. Duc and K. Kim. Securing HB^{+} against GRS man-in-the-middle attack. In *SCIS*, 2007.

[14] Jean-Bernard Fischer and Jacques Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 245–255. Springer, May 1996.

[15] Martin Fürer. Faster integer multiplication. *SIAM J. Comput.*, 39(3):979–1005, 2009.

[16] Henri Gilbert, Matt Robshaw, and Herve Sibert. An active attack against HB^{+} — A provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237, 2005. `http://eprint.iacr.org/`.

[17] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. Good variants of HB^{+} are hard to find. In Gene Tsudik, editor, *FC 2008*, volume 5143 of *LNCS*, pages 156–170. Springer, January 2008.

[18] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. $\mathsf{HB}^{\#}$: Increasing the security and efficiency of HB^{+}. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 361–378. Springer, April 2008.

[19] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33:792–807, 1986.

[20] Stefan Heyse, Eike Kiltz, Vadim Lyubashevsky, Christof Paar, and Krzysztof Pietrzak. Lapin: An efficient authentication protocol based on Ring-LPN. In Anne Canteaut, editor, *FSE*, volume 7549 of *Lecture Notes in Computer Science*, pages 346–365. Springer, 2012.

[21] Nicholas J. Hopper and Manuel Blum. Secure human identification protocols. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 52–66. Springer, December 2001.

[22] Ari Juels and Stephen A. Weis. Authenticating pervasive devices with human protocols. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 293–308. Springer, August 2005.

[23] Jonathan Katz and Ji Sun Shin. Parallel and concurrent security of the HB and HB^+ protocols. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 73–87. Springer, May / June 2006.

[24] Jonathan Katz, Ji Sun Shin, and Adam Smith. Parallel and concurrent security of the HB and HB^+ protocols. *Journal of Cryptology*, 23(3):402–421, July 2010.

[25] Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998.

[26] Eike Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, and Daniele Venturi. Efficient authentication from hard learning problems. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 7–26. Springer, 2011.

[27] Éric Levieil and Pierre-Alain Fouque. An improved LPN algorithm. In Roberto De Prisco and Moti Yung, editors, *SCN 06*, volume 4116 of *LNCS*, pages 348–359. Springer, September 2006.

[28] Vadim Lyubashevsky and Daniel Masny. Man-in-the-middle secure authentication schemes from LPN and weak PRFs. *IACR Cryptology ePrint Archive*, 2013:92, 2013.

[29] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

[30] Jorge Munilla and Alberto Peinado. HB-MP: A further step in the HB-family of lightweight authentication protocols. *Computer Networks*, 51(9):2262–2267, 2007.

[31] Khaled Ouafi, Raphael Overbeck, and Serge Vaudenay. On the security of $\mathsf{HB}^\#$ against a man-in-the-middle attack. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 108–124. Springer, December 2008.

[32] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.

[33] Krzysztof Pietrzak. Subspace LWE. In Ronald Cramer, editor, *TCC*, volume 7194 of *Lecture Notes in Computer Science*, pages 548–563. Springer, 2012.

[34] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[35] Schönhage and V. Strassen. Schnelle multiplikation grosser zahlen. *Computing*, 7, 1971.

[36] Jeroen Van De Graaf. *Towards a formal definition of security for quantum protocols*. PhD thesis, Universite de Montreal, Monreal, P.Q., Canada, Canada, 1998. AAINQ35648.

$\mathcal{P}(\mathbf{S}, \tau, n_r, n_s)$		$\mathcal{V}(\mathbf{S}, \tau', n_r, n_s)$
	$\xleftarrow{\mathbf{v}}$	$\mathbf{v} \stackrel{\$}{\leftarrow} \{\mathbf{x} \in \mathbb{Z}_2^{2\ell} : \mathbf{wt}(\mathbf{x}) = \ell\}$
if $\mathbf{wt}(\mathbf{v}) \neq \ell$ abort		
$\mathbf{R} \stackrel{\$}{\leftarrow} \mathbb{Z}_2^{\ell \times n_r}$; $\mathbf{e} \stackrel{\$}{\leftarrow} \mathsf{Ber}_\tau^{n_r \times n_s}$		
$\mathbf{z} := \mathbf{R}^\mathsf{T} \cdot \mathbf{S}_{\downarrow \mathbf{v}} + \mathbf{e}$	$\xrightarrow{\mathbf{z},\mathbf{R}}$	if $\mathsf{rank}(\mathbf{R}) \neq n_r$ reject
		if $\mathbf{wt}(\mathbf{R}^\mathsf{T} \cdot \mathbf{S}_{\downarrow \mathbf{v}} + \mathbf{z}) > n_r \cdot n_s \cdot \tau'$ reject else accept

Figure 9: A generalization of the protocol from Figure 6 where we trade a larger key (which now is a matrix $\mathbf{S} \in \mathbb{Z}_2^{2\ell \times n_s}$) for lower communication and randomness complexity. The protocol is as secure as the protocol from Figure 6 (wich is the special case where $n_r = n$ and $n_s = 1$) with $n = n_r \cdot n_s$.

[37] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, May 2005.

[38] John Watrous. Zero-knowledge against quantum attacks. *SIAM J. Comput.*, 39(1):25–58, 2009.

A Extensions

In this section we discuss some extensions of the protocols we presented in Section 3 and Section 4.

A.1 Trading Key-Size for Communication Complexity

A disadvantage of the schemes proposed in this paper is their large communication complexity. For example, in the authentication protocol from Section 3 the prover has to send the entire $\ell \times n$ matrix $\mathbf{R}$ to the verifier. Similarly, in the MACs from Section 4, the tag is computed by permuting a string of the form $(\mathbf{R}, \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{m}) + \mathbf{e}, \mathbf{b})$, where again $\mathbf{R}$ is an $\ell \times n$ matrix.

We now explain a simple tradeoff that is originally due to Gilbert *et al.* [18]. Consider the authentication protocol from Section 3. Let $1 \leq c \leq n$ be an integer parameter and let $n_s := c$ and $n_r := n/c$. The idea is to use a larger secret matrix $\mathbf{S} \in \mathbb{Z}_2^{2\ell \times n_s}$ (instead of just one vector $\mathbf{s}$) and a smaller random matrix $\mathbf{R} \in \mathbb{Z}_2^{\ell \times n_r}$ (instead of $\mathbf{R} \in \mathbb{Z}_2^{\ell \times n}$). The resulting protocol is illustrated in Figure 9. Similar extensions can be easily derived for the MACs of Section 4, where the tradeoff is more important due to the pairwise independent permutation π which is the computational bottleneck of the protocol. See Figure 3 for a comparison of the resulting complexities. The proof of Theorem 3.1, Theorem 4.1 and Theorem 4.8 can be adapted to show the same security and completeness results.

A.2 An alternative Two-Round Authentication Protocol

In this section we describe an alternative 2-round authentication protocol and sketch the proof of its active security under the hardness of the $\mathsf{SLPN}_{\tau,\ell,\ell}$ problem. The difference with the scheme from

Section 3 is the way the session key $\mathbf{s}(\mathbf{v})$ is computed. Whereas in the AUTH protocol from Figure 5 the session key is computed as $\mathbf{s}(\mathbf{v}) = \mathbf{s}_{\downarrow\mathbf{v}}$, in the new protocol it is computed as $\mathbf{s}(\mathbf{v}) = \mathbf{M_v}\mathbf{s}_0 + \mathbf{s}_1$, where $\mathbf{M_v} \in \mathbb{Z}_2^{\ell\times\ell}$ is the matrix representation of a finite field multiplication with $\mathbf{v}$ (see definition below), and $(\mathbf{s}_0, \mathbf{s}_1) \in \mathbb{Z}_2^{\ell}$ is the secret key.

Definition A.1. For $\mathbf{c} \in \mathbb{Z}_2^{\ell}$, let $\mathbf{M_c} \in \mathbb{Z}_2^{\ell\times\ell}$ denote the matrix of the linear map implementing the finite field multiplication with $\mathbf{c}$ when interpreted as an element in $\mathbb{F}_{2^\ell}$.[10]

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

$$\text{for all distinct vectors } \mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^{\ell}, \quad \mathbf{M_a} - \mathbf{M_b} \text{ is an invertible matrix.} \tag{A.1}$$

The mapping $\varphi(\mathbf{c}) = \mathbf{M_c}$ is called encoding with full-rank differences in [8]. An explicit construction was given in [11].

We are now ready to define the modified authentication protocol.

- 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 $n \le \ell/2$)

- 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 2-round authentication protocol with prover $\mathcal{P}_{\tau,n}$ and verifier $\mathcal{V}_{\tau',n}$ is given in Figure 5.

Prover $\mathcal{P}_{\tau,n}((\mathbf{s}_0, \mathbf{s}_1) \in \mathbb{Z}_2^{\ell})$		Verifier $\mathcal{V}_{\tau',n}((\mathbf{s}_0, \mathbf{s}_1) \in \mathbb{Z}_2^{\ell})$
	$\xleftarrow{\mathbf{v}}$	$\mathbf{v} \xleftarrow{\$} \mathbb{Z}_2^{\ell}$
$\mathbf{R} \xleftarrow{\$} \mathbb{Z}_2^{\ell\times n}$; $\mathbf{e} \xleftarrow{\$} \mathsf{Ber}_\tau^n$ $\mathbf{s}(\mathbf{v}) := \mathbf{M_v}\mathbf{s}_0 + \mathbf{s}_1$.		
$\mathbf{z} := \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{v}) + \mathbf{e} \in \mathbb{Z}_2^n$	$\xrightarrow{(\mathbf{R},\mathbf{z})}$	if $\mathsf{rank}(\mathbf{R}) \neq n$ reject if $\mathbf{wt}(\mathbf{z} + \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{v})) > n \cdot \tau'$ reject, else accept

Figure 10: Two-round protocol AUTH_2 with active security from the LPN assumption.

Even though the protocol is less efficient than AUTH, it has a considerably simpler proof and can give intuition for MAC_2.

We now sketch the reduction from the $\mathsf{SLPN}_{\tau,\ell,\ell}$ 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

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

[10] This 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 (A.1). 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 $\mathsf{SLPN}(\mathbf{M_v} - \mathbf{M_{v^*}}, \mathbf{c})$. 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.

A.3 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 [34]. The most appealing characteristic of this problem is that it enjoys for certain parameters a worst-case hardness guarantee [34, 32]. We informally recall the LWE problem below. Let $q \geq 2$ be a prime and denote with $\mathsf{Gau}_{q,\tau}$ 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 $\lfloor q \cdot x \rceil \bmod q$. For a random secret $\mathbf{s} \in \mathbb{Z}_q^\ell$, the (decisional) $\mathsf{LWE}_{q,\tau,\ell}$ problem is to distinguish samples of the form $(\mathbf{r}, \mathbf{r}^\mathsf{T} \cdot \mathbf{s} + e)$ from uniformly random samples in $\mathbb{Z}_q^\ell \times \mathbb{Z}_q$, where $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^\ell$, $e \xleftarrow{\$} \mathsf{Gau}_{q,\tau}$ 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 [33] 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 $\mathsf{Gau}_{q,\tau}$ 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 $\mathbf{wt}(\cdot)$. Given a vector $\mathbf{e} \in \mathbb{Z}_q^n$ sampled from $\mathsf{Gau}_{q,\tau}^n$ (where $\mathbf{e}$ has the form $\mathbf{z} - \mathbf{R}^\mathsf{T} \cdot \mathbf{s}_{\downarrow \mathbf{v}} \bmod q$ for an honest execution of the protocol from Section 3 or $\mathbf{z} - \mathbf{R}^\mathsf{T} \cdot \mathbf{s}(\mathbf{v}) \bmod q$ for the schemes from Section 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 $n \left\lfloor \frac{q}{2} \right\rfloor \cdot \tau'$ (which will happen with overwhelming probability by the standard tail bound on Gaussians).

The change of 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 computational complexity, which may be a problem in the context of resource bounded devices.

B A Technical Lemma

Lemma B.1. *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*

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

Proof. Assume we sample the n columns of a matrix $\mathbf{M} \in \mathbb{Z}_2^{n+d \times n}$ one by one. For $i = 1, \ldots, n$ 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 ith 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

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

□