Is Public-Key Encryption Based on LPN Practical?

Ivan Damgård[*] Sunoo Park[†]

Abstract

We conduct a study of the cryptosystem suggested by Alekhnovich based on the Learning Parity with Noise (LPN) problem. We consider several improvements to the scheme, inspired by similar existing variants of Regev's LWE-based cryptosystem, and compute which parameters one would need for various security levels, given the currently best known attacks. Our conclusion is that LPN-based public-key cryptography is in several respects not competitive with existing schemes, as both public key, ciphertext and encryption time become very large already for 80-bit security. There are ways to overcome some of these problems, but we do not know how to make both public key and ciphertext be small at the same time. On the other hand, decryption time seems to be competitive with existing schemes that use exponentiation to decrypt, if more than 128-bit security is desired. Thus LPN based public key encryption only seems attractive if one has a very specialized application where the time spent on decryption can be considered the only bottleneck.

1 Introduction

The decisional LPN problem is that of distinguishing from random a set of samples, each of the form $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle \oplus e)$, where $\mathbf{a} \in \mathbb{Z}_2^n$ is uniformly random (for some parameter $n \in \mathbb{N}$), $e \leftarrow \mathsf{Ber}_\tau$ where Ber_τ denotes the Bernoulli distribution (with some parameter $\tau \in \mathbb{R}$), and $\mathbf{s} \in \mathbb{Z}_2^n$ is a random secret fixed over all samples. In the search version of the problem, the goal is to find the secret vector $\mathbf{s}$. A more detailed definition of the problem is given in Section 2. Note that adding the "Bernoulli noise" is essential to make the problem non-trivial, since otherwise the secret can be easily found by Gaussian elimination given $O(n)$ samples.

LPN samples are computationally very simple to generate, but the problem nevertheless seems to be very hard. The two main types of non-trivial attack on LPN are exhaustive search over possible error vectors, and series of attacks based on the Blum-Kalai-Wasserman (BKW) algorithm [BKW03]. The original BKW algorithm was estimated to have slightly subexponential time complexity of $2^{O(n/\log n)}$ for $2^{O(n/\log n)}$

[*]Department of Computer Science, Aarhus University, Aarhus, Denmark.
[†]Computer Laboratory, University of Cambridge, Cambridge, UK.

samples. Subsequent work by Lyubashevsky gave a variant algorithm with runtime $2^{O(n/\log\log n)}$ for $n^{1+\epsilon}$ samples [Lyu05]. A further modification proposed more recently by Kirchner [Kir11] achieved similar runtimes with $O(n)$ samples. Practical implementations of optimised variants of the above algorithms were done by Levieil and Fouque [LF06] and Bernstein and Lange [BL12].

The computational simplicity of LPN make it very attractive for cryptographic applications, and indeed, many applications of the "symmetric crypto" type have been suggested [GRS08; JW05]. Doing public-key cryptography based on LPN seems to be much harder; however, in [Ale03], Alekhnovich suggested a public-key cryptosystem based on a variant of decisional LPN, where τ is not constant as in standard LPN but decreases with increasing n – in fact, $\tau \approx 1/\sqrt{n}$. This problem might be easier than LPN with constant τ, but no separation between the problems in the sense of asymptotic complexity is known.

In this paper we study a variant of Alekhnovich's original cryptosystem, which is favourable for analysis as well as practical efficiency reasons. A basic version of this scheme was first communicated to us by Cash [Cas12], and seems to be folklore, at least in some parts of the community, but we were not able to find any published record of it. It is similar in structure to Regev's cryptosystem based on the hardness of the Learning With Errors (LWE) problem [Reg05]. We are therefore able to improve it to get a better plaintext to ciphertext size ratio in a way similar to a corresponding improvement by Peikert *et al.* of Regev's scheme [PVW07]. Finally, we add a further slight optimisation to the improved scheme by the use of all-or-nothing transforms, which yields a noticeable advantage in terms of practical parameters. The idea behind the proofs of security we give can be traced to an invited talk given by Micciancio [Mic10] (although the reader should be aware that the talk was primarily about encryption based on LWE).

The question we ask ourselves in this paper is as follows: given what we know about LPN, how (un)attractive is public-key cryptography based on LPN as an alternative to more well known cryptosystems *in practice*?

We are not aware of any previous attempts to figure out a precise answer to this. It seems that the general perception among cryptographers has been that LPN-based public-key must "of course" be totally impractical: Alekhnovich's version of the LPN problem seems to be easier than standard LPN (due to the limitation on the noise rate) so to ensure security would require huge values of n that would render the whole scheme impracticable. However, it is important to consider that for a practical application of an LPN-based scheme, one must choose concrete values of parameters n and τ, and what then matters is not the asymptotic complexity of solving the underlying problem, but whether those concrete values are vulnerable to attack by state-of-the-art algorithms.

In the final part of this paper, we consider how the most recent attacks known would perform against the LPN instances used in the LPN-based cryptosystem we propose. We set as a goal to have a 25% probability of incorrect decryption of a bit sent, which means that we would need error correction with an expansion factor of about 5 to correct these errors. As examples we find that for 80-bit security we need $n = 9000$ whereas for 128 bit security we need $n = 29000$.

This means that public keys will be very large, several megabyte already for 80-bit security, and in the basic scheme, a single bit of ciphertext is expanded to a ciphetext of length n (essentially). There are partial ways of getting around these problems: in the basic scheme, the largest part of the public key is a matrix of size (essentially) n by $2n$ bits, this matrix is random and could therefore be pseudorandomly generated on the fly instead of being stored, furthermore it does not have to user specific. As for the ciphertext to plaintext ratio, this can be reduced by going to the improved scheme. In this scheme, however, the public key consists of two large matrices, of which only one is random and non user-specific. Therefore, unfortunately, we do not know how to reduce public-key size and ciphertext size at the same time.

As for computing time, the big theoretical advantage of LPN is that the time spent per bit decrypted is linear in n, in contrast to well-known factoring and discrete log based schemes where this time is quadratic. Of course, it is not clear that this advantage will actually materialize for practical values of the parameters. To get a feeling for this, we compared LPN performance to RSA for various security levels and computed the time one would need for a typical application, where for k bits of security, one wants to send a k-bit symmetric key using public-key encryption. For instance, for $k = 128$ the decryption time for our LPN implementation would be 50 msec. We found that RSA outperforms LPN for the security levels we considered, but there was a clear tendency to the advantage of LPN for increasing levels of security. Namely, for $80, 112$ and 128 bits of security, LPN was slower than RSA by factors of $43, 22$ and 5.7 respectively. So if one wants to go above 128 bits of security, it seems that LPN would be competitive, but of course only if decryption time was the only bottleneck.

LPN encryption is slower than decryption because one needs a full scale matrix with vector product. So of course RSA encryption is dramatically faster than LPN. On the other hand, if preprocessing is possible, then the encryption time for LPN can be reduced to essentially the same time as decryption.

We did not compare to times for elliptic curve cryptography (ECC). We expect, however, that LPN decryption will be less competitive here because keys for ECC do not have to grow as fast with increasing security as in RSA.

Finally, one should note that if one desires security against quantum attacks, then neither RSA nor ECC are secure, and one should instead compare to LWE based cryptosystems, but this is not in scope of this paper.

2 The Cryptosystem

We begin by establishing some notational conventions that shall be used in this paper, and providing a more formal definition of the LPN problem.

Notation. Ber_τ denotes the Bernoulli distribution with parameter τ. Ber_τ^k denotes the distribution of vectors in $\mathbb{Z}_2^k$ where each entry of the vector is drawn independently from Ber_τ. $\mathsf{Bin}_{n,\tau}$ denotes the binomial distribution with n trials, each with success probability τ. Where it is clear from context, we sometimes use the term "indistinguishable" in lieu of "computationally indistinguishable". $\oplus$ denotes the exclusive-or operation.

Definition 2.1. (Decisional LPN problem)
Take parameters $n \in \mathbb{N}$ and $\tau \in \mathbb{R}$ with $0 < \tau < 0.5$ (the noise rate). A distinguisher D *is said to (q, t, ε)-solve the decisional* $\mathsf{LPN}_{n,\tau}$ *problem if*

$$\left| \Pr_{\mathbf{s},\mathbf{A},\mathbf{e}}[\mathsf{D}(\mathbf{A}, \mathbf{As} \oplus \mathbf{e}) = 1] - \Pr_{\mathbf{r},\mathbf{A}}[\mathsf{D}(\mathbf{A}, \mathbf{r}) = 1] \right| \geq \varepsilon$$

where $\mathbf{s} \in_R \mathbb{Z}_2^n$, $\mathbf{A} \in_R \mathbb{Z}_2^{q\times n}$, and $\mathbf{r} \in_R \mathbb{Z}_2^q$ are uniformly random and $\mathbf{e} \leftarrow \mathsf{Ber}_\tau^q$, and the distinguisher runs in time at most t.

Somewhat unusually, the decisional and search variants of the LPN problem are polynomially equivalent, meaning that the existence of an attack requiring q samples against decisional LPN implies the existence of an attack against search LPN requiring polynomial in q samples. More precisely:

Lemma 2.2. (Lemma 1 from [KSS10]) *If there exists a distinguisher* D *that (q, t, ε)-solves the decisional* $\mathsf{LPN}_{n,\tau}$ *problem, then there exists a distinguisher* D' *that (q', t', ε')-solves the search* $\mathsf{LPN}_{n,\tau}$ *problem where $q' = O(q \log n/\varepsilon^2)$, $t' = O(tn \log n/\varepsilon^2)$, and $\varepsilon' = \varepsilon/4$.*

The hardness assumptions used in this paper are based on the decisional LPN problem; henceforth, the term "LPN problem" shall refer to the decisional variant.

As usual, a probability $\varepsilon(n)$ is said to be negligible if $\varepsilon(n) \leq 1/p(n)$ for any polynomial p and all large enough n. Using this, we can state the computational assumption we will base the cryptosystem on:

Definition 2.3. (Decisional LPN assumption, DLPN)
Assume probabilistic algorithm D (q, t, ε)-solves the decisional $\mathsf{LPN}_{n,\tau}$ *problem for all large enough n, where τ is $\Theta(1/\sqrt{n})$, t is polynomial in n and q is $O(n)$. Then ε is negligible as a function of n.*

Adopting the standard notion of computational indistinguishability, this can be equivalently stated as the assumption that $(\mathbf{A}, \mathbf{As} \oplus \mathbf{e})$ is computationally indstinguishable from $(\mathbf{A}, \mathbf{r})$ with the choices of τ and q we made above.

We define the basic LPN cryptosystem as follows.

Definition 2.4. (Basic LPN cryptosystem)
The key generation, encryption, and decryption functions of the basic LPN cryptosystem are given below. The parameters are $n \in \mathbb{N}$, the length of the secret, and $\tau \in \mathbb{R}$, the noise rate. All operations are performed over $\mathbb{Z}_2$.

- $\mathsf{BasicLPNKeyGen}()$*: Choose a secret key $\mathbf{s} \in \mathbb{Z}_2^n$ uniformly at random. For the public key, choose a matrix $\mathbf{A} \in \mathbb{Z}_2^{(2n+2)\times n}$ uniformly at random, and choose an error vector $\mathbf{e} \in \mathbb{Z}_2^{2n+2}$ according to Ber_τ^{2n+2}. Then the public key is the pair $(\mathbf{A}, \mathbf{b})$, where $\mathbf{b} = \mathbf{As} + \mathbf{e}$.*

- $\mathsf{BasicLPNEnc}(pk = (\mathbf{A}, \mathbf{b}), v)$*: To encrypt a message bit* $v \in \mathbb{Z}_2$*, choose a vector* $\mathbf{f} \in \mathbb{Z}_2^{2n+2}$ *according to* Ber_τ^{2n+2}*. Then the ciphertext is the pair* $(\mathbf{u}, c)$ *where* $\mathbf{u} = \mathbf{f}^T\mathbf{A}$ *and* $c = \mathbf{f}^T\mathbf{b} + v$*.*

- $\mathsf{BasicLPNDec}(sk = \mathbf{s}, (\mathbf{u}, c))$*: The decryption is* $d = c + \langle \mathbf{u}, \mathbf{s} \rangle$*.*

Remark. In [DMQN12], Döttling, Müller-Quade, and Nascimento present an LPN-based CPA-secure public key encryption scheme similar to the above. The difference is that their encryption process outputs a pair of the form $(\mathbf{u} + \mathbf{e}_1, c + e_2)$, where $\mathbf{u}$ and c are defined as above, and $\mathbf{e}_1 \sim \mathsf{Ber}_\tau^n$ and $e_2 \sim \mathsf{Ber}_\tau$ are chosen during encryption. This yields a higher decryption error than for the scheme presented in this paper and so, considering that the hardness assumption upon which the two cryptosystems are based is the same (that of the LPN problem), the scheme presented here seems to compare favourably.

We now prove correctness and security for the basic LPN cryptosystem. Some supporting lemmas are needed.

Lemma 2.5. *Let* $X \sim \mathsf{Bin}_{n,\tau}$*. Then the probability that* X *is even is* $\frac{1}{2} + \frac{(1-2\tau)^n}{2}$*.*

Proof. The probability generating function of X is

$$G_X(z) = \sum_{k=0}^{n} z^k \Pr[X = k] = ((1-\tau) + \tau z)^n.$$

Define $G(z) = \frac{1}{2}(G_X(z) + G_X(-z))$. Then since terms with odd powers cancel out,

$$G(z) = \sum_{k=0}^{n} z^{2k} \Pr[X = 2k],$$

so $G(1)$ is equal to the total probability that X takes an even value:

$$\Pr[X \text{ is even}] = G(1) = \frac{1}{2}(G_C(1) + G_C(-1)) = \frac{1}{2} + \frac{(1-2\tau)^n}{2}.$$

□

Lemma 2.6. *For any* k *such that* $\lim_{n\to\infty} \frac{n}{k} = \infty$*, it holds that* $\lim_{n\to\infty}(1 + \frac{k}{n})^n = e^k$*.*

Proof. Take any k such that $\lim_{n\to\infty} \frac{n}{k} = \infty$. Then:

$$\lim_{n\to\infty} (1 + \frac{k}{n})^n = \lim_{\frac{n}{k}\to\infty} (1 + \frac{k}{n})^{\frac{n}{k}\cdot k} = \lim_{n'\to\infty} (1 + \frac{1}{n'})^{n'\cdot k} = e^k.$$

□

Lemma 2.7. (Correctness) *For any constant* $\varepsilon > 0$*, it holds that* τ *can be chosen with* $\tau = \Theta(\frac{1}{\sqrt{n}})$ *such that the probability of correct decryption by* $\mathsf{BasicLPNDec}$ *is at least* $1 - \varepsilon$*.*

Proof. The decrypted bit d is equal to the correct plaintext v if and only if $\mathbf{f}^T\mathbf{e} = 0$, since

$$d = c + \mathbf{s}^T\mathbf{u} = \mathbf{f}^T\mathbf{b} + v + \mathbf{s}^T\mathbf{f}^T\mathbf{A} = \mathbf{f}^T(\mathbf{A}\mathbf{s} + \mathbf{e}) + v + \mathbf{s}^T\mathbf{f}^T\mathbf{A} = \mathbf{f}^T\mathbf{e} + v.$$

Let e_i and f_i denote the entries of $\mathbf{e}$ and $\mathbf{f}$ respectively. Define $C_i = e_i \cdot f_i$. These C_i are independent and identically distributed with the distribution Ber_{τ^2}. Let $C = \sum_i C_i$. Then $C \sim \mathsf{Bin}_{2n+2,\tau^2}$.

Observe that $\mathbf{f}^T\mathbf{e} = 0$ if and only if C takes an even value. From Lemma 2.5, then, $\Pr[\mathbf{f}^T\mathbf{e} = 0] = \frac{1}{2} + \frac{(1-2\tau^2)^{2n+2}}{2}$. Take $0 < \tau \leq O(\frac{1}{\sqrt{n}})$: for τ in this range, $\tau^2 n = O(1)$, so $\lim_{n\to\infty} \frac{n}{\tau^2 n} = \infty$. Applying Lemma 2.6 yields:

$$\lim_{n\to\infty} (1 - 2\tau^2)^{2n+2} = \lim_{n\to\infty} (1 - \frac{2\tau^2(2n+2)}{2n+2})^{2n+2} = e^{-2\tau^2(2n+2)}.$$

Hence, for large n,

$$\Pr[\mathbf{f}^T\mathbf{e} = 0] = \frac{1}{2} + \frac{(1-2\tau^2)^{2n+2}}{2} \approx \frac{1 + e^{-2\tau^2(2n+2)}}{2}.$$

If $\tau = \frac{c}{\sqrt{n}}$ for some constant c, then the exponent $-2\tau^2(2n+2)$ of the above equation is constant. Observe that $\lim_{c\to 0} -2\tau^2(2n+2) = 0$, so $\lim_{c\to 0} \frac{1+e^{-2\tau^2(2n+2)}}{2} = 1$. It follows that for $\tau = \Theta(\frac{1}{\sqrt{n}})$, for any constant $\varepsilon > 0$, the probability of correct decryption by $\mathsf{BasicLPNDec}$ is at least $1 - \varepsilon$ provided that c is chosen sufficiently close to 0. □

Remark. Provided that the decryption error rate is low enough, error correcting codes may be employed to essentially eliminate the possibility of incorrectly received bits (in the case that we consider messages of multiple bits).

Lemma 2.8. (PSEUDORANDOM PUBLIC KEYS) *Under the DLPN assumption, the distribution of the public keys* $(\mathbf{A}, \mathbf{b})$ *generated by* $\mathsf{BasicLPNKeyGen}$ *is computationally indistinguishable from uniform over* $\mathbb{Z}_2^{(2n+2)\times n} \times \mathbb{Z}_2^{2n+2}$.

Proof. The public keys generated by $\mathsf{BasicLPNKeyGen}$ are of the form $(\mathbf{A}, \mathbf{A}\mathbf{s} \oplus \mathbf{e})$, where $\mathbf{A}$, $\mathbf{s}$, and $\mathbf{e}$ are chosen as in Definition 2.1. Since furthermore q and τ are chosen as in the DLPN assumption, the required indistinguishability follows immediately. □

Notation. For a vector $\mathbf{w}$, let w_i denote its i^{th} entry; and for a matrix $\mathbf{W}$, let $\mathbf{w}_i$ denote its i^{th} column, and let $w_{i,j}$ denote the j^{th} entry of its i^{th} row.

Lemma 2.9. *For* $m \geq dn$ *for a constant* $d > 1$, *let* $\chi_{m,n}$ *be the distribution of matrices* $\mathbf{M} \in \mathbb{Z}_2^{m\times n}$ *which are sampled by choosing the columns to be a uniformly random linearly independent set. For large* n, $\chi_{m,n}$ *is statistically indistinguishable from the uniform distribution over* $\mathbb{Z}_2^{m\times n}$.

Proof. A matrix sampled from the uniform distribution over $\mathbb{Z}_2^{m\times n}$ is (perfectly) indistinguishable from one constructed by taking n column vectors of m bits drawn uniformly from $\mathbb{Z}_2^m$, since matrix columns are independent in the former distribution. For $m \geq dn$, consider generating a matrix by drawing the columns one by one. Each time a new column is drawn, it lies outside the subspace spanned by the column vectors already drawn, except with probability exponentially small in n. Therefore, with all but negligible probability, a matrix sampled from $\mathbb{Z}^{m\times n}$ will have full rank, and the result follows. □

Lemma 2.10. *Under the DLPN assumption, for any $c \in \mathbb{N}$, $(\mathbf{R}, \mathbf{f}^T\mathbf{R})$ is computationally indistinguishable from $(\mathbf{R}, \mathbf{r})$, where $\mathbf{f} \in \mathbb{Z}_2^{2(n+c)}$ is drawn from $\mathsf{Ber}_\tau^{2(n+c)}$, $\mathbf{R} \in_R \mathbb{Z}_2^{2(n+c)\times(n+2c)}$ and $\mathbf{r} \in_R \mathbb{Z}^{n+2c}$.*

Proof. Take an LPN sample of the form $(\mathbf{A}, \mathbf{A}\mathbf{s} \oplus \mathbf{e})$ constructed as detailed in Definition 2.1, with $q = 2(n+c)$. By Lemma 2.9, this is computationally indistinguishable from $(\mathbf{A}', \mathbf{A}'\mathbf{s} \oplus \mathbf{e})$ where $\mathbf{A}' \sim \chi_{2(n+c),n}$. Let $\mathbf{H} \in \mathbb{Z}_2^{2(n+c)\times(n+2c)}$ be sampled by choosing the column vectors as a uniformly random basis for the orthogonal complement $C \subseteq \mathbb{Z}_2^{2(n+c)}$ of the columns of A'. C is determined uniformly randomly by the choice of $\mathbf{A}'$, so the distribution of $\mathbf{H}$ is computationally indistinguishable from $\chi_{2(n+c),n+2c}$. It follows, by Lemma 2.9, that $\mathbf{H}$ is indistinguishable from uniformly random.

By construction, $\mathbf{H}^T\mathbf{A} = 0$, so $\mathbf{H}^T(\mathbf{A}\mathbf{s} \oplus \mathbf{e}) = \mathbf{H}^T\mathbf{A}\mathbf{s} \oplus \mathbf{H}^T\mathbf{e} = \mathbf{H}^T\mathbf{e}$. This means that since, under DLPN, $(\mathbf{A}, \mathbf{A}\mathbf{s} \oplus \mathbf{e})$ is indistinguishable from random, so is $(\mathbf{H}, \mathbf{H}^T\mathbf{e})$. The lemma then finally follows from noting that indistinguishability from random still holds if we transpose the last component. □

Theorem 2.11. *Under the DLPN assumption, the basic LPN cryptosystem is secure against chosen plaintext attack.*

Proof. Consider an instance of the basic LPN cryptosystem with parameters n and τ, where the public key is $(\mathbf{A}, \mathbf{b})$. Given the public key, one can construct a matrix $\mathbf{R} \in \mathbb{Z}_2^{(n+1)\times(2n+2)}$ with entries as follows:

$$r_{i,j} = \begin{cases} a_{i,j} & \text{for } 1 \leq i \leq 2n+2, 1 \leq j \leq n \\ b_i & \text{for } 1 \leq i \leq 2n+2, j = n+1 \\ \text{uniformly random} & \text{otherwise} \end{cases}$$

By Lemma 2.8, $(\mathbf{A}, \mathbf{b})$ is indistinguishable from random. It follows that $\mathbf{R}$ is also indistinguishable from random.

A ciphertext of the message 0 is of the form $(\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{b})$ as defined by LPNEnc. By Lemma 2.10, $\mathbf{r} = \mathbf{f}^T\mathbf{R}$ is indistinguishable from random. Observe that the first n entries of $\mathbf{r}$ are exactly the entries of $\mathbf{f}^T\mathbf{A}$, and that $r_{n+1} = \mathbf{f}^T\mathbf{b}$. Therefore, if $(\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{b})$ were distinguishable from random, then $\mathbf{r}$ would be distinguishable from random – but this would contradict Lemma 2.10. Therefore, a ciphertext of the message 0 is indistinguishable from random.

For any ciphertext $\kappa = (\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{b} + 1)$ of the message 1, there is a corresponding ciphertext of the message 0, $\kappa' = (\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{b})$, that differs from κ only in that one bit is flipped. Furthermore, $\Pr[\mathsf{LPNEnc}(1) = \kappa] = \Pr[\mathsf{LPNEnc}(0) = \kappa']$, so a ciphertext of 1 has exactly the same distribution as a ciphertext of 0, except that the final ciphertext bit is flipped. Inverting a uniformly random bit yields a uniformly random output, and we have already established that a ciphertext of the message 0 is indistinguishable from random. Hence, a ciphertext of the message 1 is also indistinguishable from random.

Therefore, ciphertexts produced by LPNEnc are indistinguishable from random. □

The basic LPN cryptosystem can be significantly improved in efficiency by a relatively simple modification reducing the ciphertext expansion factor (the ratio of ciphertext to plaintext length) from $\tilde{O}(n)$ to as low as $O(1)$. This is achieved by re-using the encryption randomness over up to $\ell = O(n)$ public key rows. To allow for this, we require ℓ independent secret keys $\mathbf{s}_i$ and ℓ independent error vectors $\mathbf{e}_i$ – thus, the secret key size increases from $O(n)$ to $O(n^2)$, while the public key size remains asymptotically unchanged. The efficacy of the modification is based on the fact that a large part of the time taken by the original cryptosystem's operations is due to the large matrix $\mathbf{A}$.

This modification to the LPN cryptosystem is very similar in structure to the modification to Regev's LWE-based cryptosystem proposed by Peikert, Vaikuntanathan, and Waters [PVW07]. The modified cryptosystem is presented below.

Definition 2.12. (Improved LPN cryptosystem)
Parameters n and τ below are as in Definition 2.4. Additionally, we introduce $\ell = O(n)$, the length of plaintext that can be encrypted in a single operation. All operations are, as before, performed over $\mathbb{Z}_2$.

- LPNKeyGen()*: Choose a secret key $\mathbf{S} \in \mathbb{Z}_2^{n\times\ell}$ uniformly at random. For the public key, choose a matrix $\mathbf{A} \in \mathbb{Z}_2^{2(n+\ell)\times n}$ uniformly at random, and choose an error matrix $\mathbf{E} \in \mathbb{Z}_2^{2(n+\ell)\times\ell}$ according to $\mathsf{Ber}_\tau^{2(n+\ell)\times\ell}$. Then the public key is the pair $(\mathbf{A}, \mathbf{B})$, where $\mathbf{B} = \mathbf{AS} + \mathbf{E}$.*
- LPNEnc$(pk = (\mathbf{A}, \mathbf{B}), \mathbf{v})$*: To encrypt a message $\mathbf{v} \in \mathbb{Z}_2^\ell$, choose a vector $\mathbf{f} \in \mathbb{Z}_2^{2(n+\ell)}$ according to $\mathsf{Ber}_\tau^{2(n+\ell)}$. Then the ciphertext is the pair $(\mathbf{u}, \mathbf{c})$ where $\mathbf{u} = \mathbf{f}^T\mathbf{A}$ and $\mathbf{c} = \mathbf{f}^T\mathbf{B} + \mathbf{v}^T$.*
- LPNDec$(sk = \mathbf{S}, (\mathbf{u}, \mathbf{c}))$*: The decryption is $\mathbf{d}$ such that $\mathbf{d}^T = \mathbf{c} + \mathbf{S}^T\mathbf{u}$.*

We now prove correctness and security for the improved cryptosystem.

Lemma 2.13. (Correctness) *For each fixed choice of $\mathbf{f}$, encryption followed by decryption of a message $\mathbf{v}$ in the improved LPN cryptosystem is equivalent to sending each bit of $\mathbf{v}$ through a binary symmetric channel with some error probability ρ. Furthermore, for any constant $\varepsilon > 0$, τ can be chosen with $\tau = \Theta(\frac{1}{\sqrt{n}})$ such that, except with negligible probability, $\rho \leq \varepsilon$.*

Proof. The decryption is equal to the correct plaintext $\mathbf{v}$ if and only if $\mathbf{f}^T\mathbf{E} = \mathbf{0}$ (where $\mathbf{0}$ denotes the zero vector in $\mathbb{Z}_2^\ell$), since

$$\mathbf{d}^T = \mathbf{c} + \mathbf{S}^T\mathbf{u} = \mathbf{f}^T\mathbf{B} + \mathbf{v}^T + \mathbf{S}^T\mathbf{f}^T\mathbf{A} = \mathbf{f}^T(\mathbf{A}\mathbf{S} + \mathbf{E}) + \mathbf{v}^T + \mathbf{S}^T\mathbf{f}^T\mathbf{A} = \mathbf{f}^T\mathbf{E} + \mathbf{v}^T.$$

Let $|\mathbf{f}|$ denote the Hamming weight of $\mathbf{f}$. For any given $\mathbf{f}$, the independence of the columns $\mathbf{e}_i$ of $\mathbf{E}$ implies that the transmitted bits do indeed go independently through a noisy channel, which has error probability determined by $|\mathbf{f}|$. By Lemma 2.5, for any given weight $|\mathbf{f}|$, it holds that $\Pr[\mathbf{f}^T\mathbf{e}_i = 0] = \frac{1}{2} + \frac{(1-2\tau)^{|\mathbf{f}|}}{2}$. Observe that this probability increases as $|\mathbf{f}|$ decreases.

Let A denote the event that $|\mathbf{f}| \le 3(n+\ell)\tau$. $|\mathbf{f}|$ is distributed according to $\mathsf{Bin}_{2(n+\ell),\tau}$. By a Chernoff bound,

$$\Pr[A] < 1 - \left(\frac{\sqrt{e}}{1.5\sqrt{1.5}}\right)^{2(n+\ell)\tau}.$$

Note that $\left(\frac{\sqrt{e}}{1.5\sqrt{1.5}}\right)^{2(n+\ell)\tau}$ is exponentially small in $(n+\ell)\tau$, and that $(n+\ell)\tau \to \infty$ as $n \to \infty$, so A occurs with overwhelming probability.

If A occurs, then the encryption followed by decryption of ℓ bits in the improved LPN cryptosystem is equivalent to sending the bits through a binary symmetric channel with error probability ρ which is at most that of the case where $|\mathbf{f}| = \lfloor 3(n+\ell)\tau \rfloor$. From Lemma 2.7, it follows that for any constant $\varepsilon > 0$, τ can be chosen subject to $\tau = \Theta(\frac{1}{\sqrt{n}})$ such that $\rho \le \varepsilon$. □

Lemma 2.14. (Pseudorandom public keys) *If the LPN problem is hard, then the distribution of the public keys* $(\mathbf{A}, \mathbf{B})$ *generated by* $\mathsf{LPNKeyGen}$ *is computationally indistinguishable from uniform over* $\mathbb{Z}_2^{2(n+\ell)\times n} \times \mathbb{Z}_2^{2(n+\ell)\times \ell}$.

Proof. We define hybrid distributions $H_0, \cdots, H_\ell$ over matrices $(\mathbf{A}, \mathbf{B}) \in \mathbb{Z}_2^{2(n+\ell)\times n} \times \mathbb{Z}_2^{2(n+\ell)\times \ell}$ such that in distribution H_k, the matrix $\mathbf{A}$ and the first k columns of $\mathbf{B}$ are uniformly randomly chosen, and the other columns of $\mathbf{B}$ are chosen according to the procedure for generating a column of $\mathbf{B}$ given by $\mathsf{LPNKeyGen}$ (for parameters n and τ). Then H_0 is exactly the distribution of the public keys generated by $\mathsf{LPNKeyGen}$, and H_ℓ is completely uniform over $\mathbb{Z}_2^{2(n+\ell)\times n} \times \mathbb{Z}_2^{2(n+\ell)\times \ell}$.

For any $k \in \{0, \cdots, \ell - 1\}$, we define a simulator $\mathcal{S}_k$ which has access to an oracle $\mathcal{O}$ that returns samples in $\mathbb{Z}_2^{2(n+\ell)\times n} \times \mathbb{Z}_2^{2(n+\ell)}$ that are either chosen uniformly at random, or are of the form $(\mathbf{A}, \mathbf{A}\mathbf{s} \oplus \mathbf{e})$ as specified in the LPN problem (in Definition 2.1). $\mathcal{S}_k$ outputs a pair $(\mathbf{A}, \mathbf{B}) \in \mathbb{Z}_2^{2(n+\ell)\times n} \times \mathbb{Z}_2^{2(n+\ell)\times \ell}$ constructed as follows. First, $\mathcal{O}$ is queried, yielding a sample $(\mathbf{A}', \mathbf{b}')$. Then, $\mathcal{S}_k$ sets the output matrix $\mathbf{A}$ equal to $\mathbf{A}'$; uniformly randomly chooses the first k columns of the output matrix $\mathbf{B}$; sets the column $\mathbf{b}_{k+1}$ equal to $\mathbf{b}'$; and for all $k < j \le \ell$, $\mathcal{S}_k$ chooses independent secret vectors $\mathbf{s}_j \in \mathbb{Z}_2^n$ uniformly at random, and independent error vectors $\mathbf{e}_j \in \mathbb{Z}_2^{2(n+\ell)}$ according to $\mathsf{Ber}_\tau^{2(n+\ell)}$, and sets $\mathbf{b}_j = \mathbf{A}\mathbf{s}_j + \mathbf{e}_j$.

Observe that if $\mathcal{O}$ samples from the uniform distribution, then the output of $\mathcal{S}_k$ has distribution H_k, and otherwise, $\mathcal{O}$ samples from the LPN distribution, so the output of

$\mathcal{S}_k$ has distribution H_{k+1}. It follows that if $\mathsf{LPN}_{n,\tau}$ is hard, then H_k and H_{k+1} are computationally indistinguishable for all $j \in \{0, \cdots, \ell-1\}$. Therefore, H_ℓ is computationally indistinguishable from uniformly random H_0. ☐

Theorem 2.15. *If the LPN problem is hard, then the improved LPN cryptosystem is secure against chosen plaintext attack.*

Proof. Consider an instance of the improved LPN cryptosystem with parameters n and τ, with $\ell = O(n)$, where the public key is $(\mathbf{A}, \mathbf{B})$. Given the public key, one can construct a matrix $\mathbf{R} \in \mathbb{Z}_2^{2(n+\ell)\times(n+\ell)}$ with entries as follows:

$$r_{i,j} = \begin{cases} a_{i,j} & \text{for } 1 \le i \le 2n+\ell, 1 \le j \le n \\ b_i & \text{for } 1 \le i \le 2n+\ell, n+1 \le j \le n+\ell \\ \text{uniformly random} & \text{otherwise} \end{cases}$$

By Lemma 2.14, $(\mathbf{A}, \mathbf{B})$ is indistinguishable from random. Therefore, $\mathbf{R}$ is also indistinguishable from random.

A ciphertext in the improved LPN cryptosystem is of the form $(\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{B} + \mathbf{v})$ as specified in Definition 2.12. By Lemma 2.10, $\mathbf{r} = \mathbf{f}^T\mathbf{R}$ is indistinguishable from random. Observe that the first n entries of $\mathbf{r}$ are exactly the entries of $\mathbf{f}^T\mathbf{A}$, and that the remaining entries of $\mathbf{r}$ are exactly those of $\mathbf{f}^T\mathbf{B}$. Hence, if $(\mathbf{f}^T\mathbf{A}, \mathbf{f}^T\mathbf{B})$ were distinguishable from random, then $\mathbf{r}$ would be distinguishable from random – but this would contradict Lemma 2.10. It follows that ciphertexts are indistinguishable from $(\mathbf{A}', \mathbf{b}' + \mathbf{v})$ where $\mathbf{A}' \in_R \mathbb{Z}_2^n, \mathbf{b}' \in_R \mathbb{Z}_2^\ell$ are uniformly random.

Now for any $\mathbf{v}, \mathbf{v}' \in \mathbb{Z}_2^\ell$, given any ciphertext $(\mathbf{u}, \mathbf{c})$ of the improved LPN cryptosystem:

$$\Pr[(\mathbf{u}, \mathbf{c}) \text{ is an encryption of } \mathbf{v}] = \Pr[\mathbf{b}' = v + c]$$
$$\Pr[(\mathbf{u}, \mathbf{c}) \text{ is an encryption of } \mathbf{v}'] = \Pr[\mathbf{b}' = v' + c]$$

Since $\mathbf{b}'$ is uniformly random, then, for any ciphertext and any pair of possible corresponding plaintexts $\mathbf{v}, \mathbf{v}'$, it is the case that to an adversary $\mathbf{v}$ and $\mathbf{v}'$ are equally likely to be the correct decryption. Therefore, the improved LPN cryptosystem is secure under chosen plaintext attack. ☐

Finally, to increase the security of the improved LPN cryptosystem for practical purposes, we propose an additional encryption/decryption step based on all-or-nothing transforms (AONTs). Intuitively, these are invertible transforms that are difficult to invert unless (almost) all transformed bits are known. They were introduced and formalised in [Riv97; Boy99]; a formal definition suitable for our purposes is provided below.

Definition 2.16. (All-or-nothing transform)
A transformation $f : \mathbb{Z}_2^s \to \mathbb{Z}_2^{s'}$ is called an all-or-nothing transform for ℓ missing bits *if the following properties hold:*

- *f is invertible.*

- *Both f and its inverse f^{-1} are efficiently computable (i.e. in polynomial time).*
- *Given up to $s' - \ell$ bits of $f(x)$, for any x, it is computationally infeasible to determine x.*

In the improved LPN cryptosystem as defined above, note that an attacker may learn one bit of plaintext by discovering just n bits of secret. By applying an AONT to the message prior to encryption, we may take advantage of the many independent n-bit secrets in parallel use in the improved cryptosystem. An adversary must, in the modified system, learn almost all transmitted bits in order to gain knowledge of any bit of the plaintext message, and thus the security level of the cryptosystem is increased by a factor of roughly the message length. This comes at the cost of a slightly larger payload to transmit, as AONTs incur a small message expansion factor, as well as the time taken to perform the AONT and inverse.

Concretely, we take Optimal Asymmetric Encryption Padding (OAEP) [BR94] as the AONT for the improved LPN cryptosystem. Originally introduced by Bellare and Rogaway for unrelated purposes, OAEP was shown by Boyko to be a provably secure AONT in [Boy99].

Definition 2.17. (Optimal Asymmetric Encryption Padding)
For parameters n and k_0, generator $G : \mathbb{Z}_2^{k_0} \to \mathbb{Z}_2^n$, and hash function $H : \mathbb{Z}_2^n \to \mathbb{Z}_2^{k_0}$, the transform $\mathsf{OAEP} : \mathbb{Z}_2^n \times \mathbb{Z}_2^{k_0} \to \mathbb{Z}_2^{n+k_0}$ is defined as follows:

$$\mathsf{OAEP}^{G,H}(x, r) = x \oplus G(r) \,||\, r \oplus H(x \oplus G(r)),$$

where $||$ denotes concatenation.

For appropriate parameter choices for the OAEP[1], it turns out that the increase in the security level of the cryptosystem allows for significantly smaller public/private key sizes, and thus the additional AONT step is found to be worthwhile with the benefits outweighing the costs.

3 Parameters

In this section we provide an overview of the methodology and efficiency of known practical attacks on LPN, with a focus on those that led to the current state-of-the-art algorithm. This will provide context for our choice of parameters for a secure cryptosystem. We note that though some known attacks on LPN focus on breaking particular LPN-based (symmetric-key) systems, there has not been, as far as we know, significant attention given thus far to the specific challenges of breaking LPN-based public key encryption such as this paper discusses. Accordingly, we analyse general applicable attacks on the LPN problem. Given the absence of known algorithms targeting the decisional LPN problem, we consider it reasonable to base our parameter estimates on the existing attacks against search LPN.

[1]Note, in particular, that the security level k_0 of the OAEP must be at least equal to the number of bits of security we hope to gain for the cryptosystem.

3.1 Known Attacks

The earliest notable attack on LPN was the Blum-Kalai-Wasserman (BKW) algorithm [BKW03]. The BKW method is based on the idea that by carefully choosing small sets of vectors from a large set of samples and computing their exclusive-or, we may create "new" LPN samples where a single coordinate is set and all the other coordinates are null (which are slightly noisier than the original samples). With a sufficient number of such "new" samples, the secret may be computed correctly with high probability, by taking a majority vote over the "new" samples for the value of each bit in the secret vector. The BKW algorithm is estimated to have time complexity $2^{O(n/\log n)}$ for $2^{O(n/\log n)}$ samples. A subsequent variant algorithm by Lyubashevsky [Lyu05] had time complexity $2^{(n/\log\log n)}$ for $n^{1+\epsilon}$ samples.

Levieil and Fouque's LF1 and LF2 algorithms [LF06] practically but not asymptotically improve upon the above, and furthermore are the first BKW-style LPN attacks with a documented implementation. Their modification is to the final step of the BKW algorithm, where instead of solving equations over one bit as in the original version, they solve equations over $b > 1$ bits at a time, with the help of Walsh-Hadamard transforms. LF2, unlike LF1, makes use of heuristics in this final step, which invalidate the upper bound on the probability of failure which is proven for LF1; however, the implementation of [LF06] did not show a difference in the success rate of the two algorithms.

More recently, Kirchner proposed a modified algorithm [Kir11] using ideas from all the prior work, that greatly decreases (to $O(n)$) the number of samples needed for a successful attack. The insight of Kirchner was to essentially reduce a standard LPN problem to an easier one, in the following way. Given a set of q LPN samples of the form $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s}\rangle \oplus \mathbf{e}_i)$ for $i \in \{1, \dots, q\}$, we may choose a set T containing n elements each in $\{1, \dots, q\}$, and assemble the vectors $\mathbf{a}_t$ for $t \in T$ into the rows of an $n \times n$ matrix $\mathbf{A}_T$. Let $\widetilde{\mathbf{e}}_T$ denote the vector whose entries are the bits $b_t = \langle \mathbf{a}_t, \mathbf{s}\rangle \oplus \mathbf{e}_t$ for $t \in T$.

Suppose we choose some random set S and hope that $\mathbf{A}_S$ is invertible. This happens with about 30% probability; as long as there are enough samples, we may re-sample S until invertibility holds. Then for any $T \neq S$ we may compute the following:

$$(\mathbf{A}_T\mathbf{A}_S^{-1}, (\mathbf{A}_T\mathbf{A}_S^{-1})(\mathbf{A}_S\mathbf{s} \oplus \widetilde{\mathbf{e}}_S) \oplus (\mathbf{A}_T\mathbf{s} \oplus \widetilde{\mathbf{e}}_T)) = (\mathbf{A}_T\mathbf{A}_S^{-1}, (\mathbf{A}_T\mathbf{A}_S^{-1})\widetilde{\mathbf{e}}_S \oplus \widetilde{\mathbf{e}}_T).$$

Notice that such newly computed pairs will have exactly the distribution of LPN samples with secret $\mathbf{e}_j$. The original secret $\mathbf{s}$ has been eliminated from the new samples, but if $\mathbf{e}_j$ is discovered, then $\mathbf{s}$ can be deduced. We have effectively converted the LPN oracle for secret $\mathbf{s}$ into one for secret $\mathbf{e}_j$ – and the new problem is much easier to solve, provided that $\mathbf{e}_j$ has low weight, as is the case in many LPN-based systems.

Bernstein and Lange's yet more recent attack [BL12] targeting the Lapin authentication protocol [Hey+12] is in fact applicable to many LPN-based systems, and is a version of Kirchner's algorithm optimised and modified for the fact that Lapin is based on ring-LPN (a variant of the LPN problem). A slight modification, mentioned briefly in [BL12], can make the attack apply also to standard LPN; this comes at a relatively minor cost in time, but the number of queries required is actually less for the LPN version.

This last attack has the best performance currently known, and therefore we shall use its timings as reference for the purposes of our parameter choices. We take the attack timings of the ring-LPN version, both because the standard LPN version is only cursorily documented in [BL12], and because we aim to take conservative parameters with a reasonable security margin against slight optimisations to the algorithm (we estimate that the timings will differ by a factor of less than 2^{10}).

3.2 Parameter Choices

The number of bit operations required for a successful run of the state-of-the-art attack, according to the analysis of [BL12], is equal to $2^{f(n,\tau,a,b,l,W,q)}$, where f is a function of the cryptosystem parameters n, τ and the algorithm parameters a, b, l, W, q, as follows:

$$f(n,\tau,a,b,l,W,q) = \sum_{w\geq 2} \binom{n}{w} \tau^w (1-\tau)^{n-w} + \log_2 \Bigg(12q(n^2+n) + a(q-1)n^2 + \Big((q-1)n - 2^b a\Big) \sum_{w\leq W} \left(\binom{n-ab-l}{w} w \right) + l2^l \sum_{w \leq W} \binom{n-ab-l}{w} \Bigg).$$

We choose that the probability of incorrect decryption of a bit should be 25%, in order to allow for error correction using codes with a reasonably low expansion factor of about 5; that is, we impose that

$$\frac{1}{2} - \frac{(1-2\tau^2)^{2n+2}}{2} = 0.25.$$

Given the low values of τ relative to n that result from the above condition, the expected number of nonzero bits in an error vector is very low, and therefore we consider it a reasonable choice to set $l = 1$ and $W = 1$ for a good attack. (Intuitively, the attack algorithm "hopes" that in a set of W error bits, l or fewer bits will be nonzero.) The parameter q can be a small integer, and does not greatly influence the algorithm's performance (note that in the ring-LPN version, q matters more), so we simply set it at a generous value of 20.

Having determined reasonable values for W, l, and q, we find the values of a and b that minimise n for a range of security levels. (Note that a and b are subject to a few additional restrictions detailed in [BL12]; we take these restrictions into account.) The results are given in the table below.

Security level (bits)	n	τ	a	b
80	9000	0.0044	7	14
112	21000	0.0029	7	14
128	29000	0.0024	8	16
196	80000	0.0015	8	16
256	145000	0.0011	7	17

We have taken slightly conservative parameters rather than aiming for exact security levels, since we only expect our parameter choices to be close to optimal[2]. For the same reason, we conservatively omitted from our calculation the several extra bits of security gained for the improved cryptosystem in the case that all-or-nothing transforms are used. We expect that all together, the conservative measures that we have taken give our parameter estimates a margin of safety of between 7 and 14 bits of security.

4 Implementation

We compare the performance of the basic and improved LPN cryptosystems and RSA in implementation, for different security levels. The LPN cryptosystems primarily entail manipulation of bit matrices and bit vectors of dimension $O(n)$. Our implementation of the improved system performs w operations in the parallel where w, the word size, is in our case equal to 64. Thus, one encryption/decryption operation for that scheme handles 64 plaintext bits in one go.

The implementation was written in C++. The implementations and all programs used for comparison purposes were run on the same machine, with a 2.6GHz Intel Core i5 processor with 8GB of RAM.

	Mean time per encryption (s)			Mean time per decryption (s)		
Security level (bits)	80	112	128	80	112	128
Basic LPN	0.428000	2.232500	4.195000	0.000050	0.000110	0.000150
Improved LPN	24.275000	129.780000	248.470000	0.001400	0.003200	0.005000
RSA	0.000020	0.000040	0.00012	0.000200	0.001250	0.008733

For security levels $80, 112$ and 128, we chose for RSA modulus sizes $1024, 2048$ and 4096, respectively. The RSA decryption time assumes that the standard chinese remainder optimisation is used to reduce decryption to two exponentiations on half-size numbers. The RSA encryption time assumes the public exponent is $2^{16} + 1$, a de facto standard in practice. In comparison, naive usage of the basic LPN scheme is much slower. On the other hand, the encryption time for the LPN scheme can also be reduced in several ways: for example, we expect that our implementation can be significantly optimised by more intensive parallelisation, such as multi-threading, asynchronous I/O to reduce memory requirements while maintaining efficiency, and use of SIMD instructions. Furthermore, in a practical implementation scenario, there may be idle time that can be allotted to preprocessing. In this case, since $\mathbf{f}^T\mathbf{A}$ and $\mathbf{f}^T\mathbf{b}$ (or $\mathbf{f}^T\mathbf{B}$, in the improved system) do not depend on the message to be encrypted, they could be precomputed offline. We found that more than 99% of encryption time is comprised of matrix multiplication, so such preprocessing would reduce the online encryption time to something similar to the time needed for decryption. Such preprocessing can also be done for some discrete-log based schemes, but not for RSA as far as we are aware.

[2]It may be of interest and reassurance that by using our method to find near-optimal parameters, we find attacks that are slightly better than the concrete examples given in [BL12] itself.

We see that the basic LPN scheme uses smaller time per operation than the improved scheme. However, correcting for the fact that the improved scheme does 64 plaintext bits at a time, the improved scheme is faster, but only slightly so.

To get a reasonable comparison between the LPN and RSA schemes, one may consider a typical application, namely for k-bit security to encrypt and decrypt a k-bit symmetric key. This can be done with one RSA operation. For LPN we need to consider that because of the 25% decryption error per bit we need to expand the plaintext by a factor of about $1/(1 - h(25\%)) \approx 5$ where h is the binary entropy function. So for instance for $k = 128$ the decryption time needed in the basic scheme is $0.00015 \cdot 128 \cdot 5$. In this way, one finds that decryption in the improved scheme is slower than RSA by factors of 42, 22 and 5.7 for 80, 112 and 128-bit security.

5 Conclusion

We have seen that LPN based public key encryption currently seems impractical in standard applications due to the fact that either public key or ciphertext will be very large for reasonable levels of security. Only if decryption time can be considered the only bottleneck will LPN be an alternative to consider in practice.

6 Acknowledgement

We are grateful to Dan Bernstein and Tanja Lange for information about recent attacks on LPN.

References

[Ale03] Michael Alekhnovich. "More on Average Case vs Approximation Complexity". In: *FOCS*. IEEE Computer Society, 2003, pp. 298–307. ISBN: 0-7695-2040-5.

[BR94] Mihir Bellare and Phillip Rogaway. "Optimal Asymmetric Encryption". In: *EUROCRYPT*. Ed. by Alfredo De Santis. Vol. 950. Lecture Notes in Computer Science. Springer, 1994, pp. 92–111. ISBN: 3-540-60176-7.

[BL12] Daniel J. Bernstein and Tanja Lange. "Never Trust a Bunny". In: *RFIDSec*. Ed. by Jaap-Henk Hoepman and Ingrid Verbauwhede. Vol. 7739. Lecture Notes in Computer Science. Springer, 2012, pp. 137–148. ISBN: 978-3-642-36139-5.

[BKW03] Avrim Blum, Adam Kalai, and Hal Wasserman. "Noise-tolerant learning, the parity problem, and the statistical query model". In: *J. ACM* 50.4 (2003), pp. 506–519.

[Boy99] Victor Boyko. "On the Security Properties of OAEP as an All-or-Nothing Transform". In: *CRYPTO*. Ed. by Michael J. Wiener. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 503–518. ISBN: 3-540-66347-9.

[Cas12] David Cash. Private communication. 2012.

[DMQN12] Nico Döttling, Jörn Müller-Quade, and Anderson C. A. Nascimento. "IND-CCA secure Cryptography based on a variant of the LPN Problem". To appear at AsiaCrypt 2012. 2012.

[GRS08] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. "How to Encrypt with the LPN Problem". In: *ICALP (2)*. Ed. by Luca Aceto et al. Vol. 5126. Lecture Notes in Computer Science. Springer, 2008, pp. 679–690. ISBN: 978-3-540-70582-6.

[Hey+12] Stefan Heyse et al. "Lapin: An Efficient Authentication Protocol Based on Ring-LPN". In: *FSE*. Ed. by Anne Canteaut. Vol. 7549. Lecture Notes in Computer Science. Springer, 2012, pp. 346–365. ISBN: 978-3-642-34046-8.

[JW05] Ari Juels and Stephen A. Weis. "Authenticating Pervasive Devices with Human Protocols". In: *CRYPTO*. Ed. by Victor Shoup. Vol. 3621. Lecture Notes in Computer Science. Springer, 2005, pp. 293–308. ISBN: 3-540-28114-2.

[KKS95] Jeff Kahn, János Komlós, and Endre Szemerédi. "On the Probability That a Random ± 1-Matrix Is Singular". In: *J. American Mathematical Society* 8.1 (1995), pp. 223–240.

[KSS10] Jonathan Katz, Ji Sun Shin, and Adam Smith. "Parallel and Concurrent Security of the HB and HB$^+$ Protocols". In: *J. Cryptology* 23.3 (2010), pp. 402–421.

[Kir11] Paul Kirchner. *Improved Generalized Birthday Attack*. Cryptology ePrint Archive, Report 2011/377. http://eprint.iacr.org/. 2011.

[LF06] Éric Levieil and Pierre-Alain Fouque. "An Improved LPN Algorithm". In: *SCN*. Ed. by Roberto De Prisco and Moti Yung. Vol. 4116. Lecture Notes in Computer Science. Springer, 2006, pp. 348–359. ISBN: 3-540-38080-9.

[Lyu05] Vadim Lyubashevsky. "The Parity Problem in the Presence of Noise, Decoding Random Linear Codes, and the Subset Sum Problem". In: *APPROX-RANDOM*. Ed. by Chandra Chekuri et al. Vol. 3624. Lecture Notes in Computer Science. Springer, 2005, pp. 378–389. ISBN: 3-540-28239-4.

[Mic10] Daniele Micciancio. Invited talk given at PKC '10. Slides available at http://cseweb.ucsd.edu/ daniele/papers/DualitySlides.pdf. 2010.

[PVW07] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. "A Framework for Efficient and Composable Oblivious Transfer". In: *IACR Cryptology ePrint Archive* 2007 (2007), p. 348.

[Reg05] Oded Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *STOC*. Ed. by Harold N. Gabow and Ronald Fagin. ACM, 2005, pp. 84–93. ISBN: 1-58113-960-8.

[Riv97] Ronald L. Rivest. “All-or-Nothing Encryption and the Package Transform”. In: *FSE*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 210–218. ISBN: 3-540-63247-6.