On Learning Parity with Noise in Different Noise Regimes

Alexander Burton, Steven Cheng

May 2022

Abstract

Learning parity with noise (LPN) is a post-quantum hardness assumption with parallels to learning with errors (LWE), closely related to the theory of error-correcting codes. While fewer cryptographic constructions are known from LPN than e.g. LWE, LPN promises concrete efficiency that in some cases rival dedicated constructions. In this brief survey, we look at authentication and public-key encryption as examples to see how cryptography is built from LPN with two specific parameter spaces (noise regimes).

1 Introduction

Consider the computational learning parity with noise (LPN) problem: given a uniform random matrix $\mathbf{A} \in \mathbb{Z}_2^{n \times m}$, a uniform random target vector $\mathbf{s} \in \mathbb{Z}_2^n$, and a Bernoulli random error vector $\mathbf{e} \in \mathbb{Z}_2^m$, recover $\mathbf{s}$ from $\left(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T\right)$. Decisional LPN asks only to distinguish $\left(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T\right)$ from $(\mathbf{A}, \mathbf{u})$, where $\mathbf{u}$ is uniform.

Computational LPN can be viewed as the average-case problem of decoding random linear codes over a binary symmetric channel. The problem has been studied extensively for decades – it is known to be NP-hard, and is generally believed to be difficult on average [HB001]. The best known attacks are exponential, and there is good evidence that quantum algorithms cannot reduce these attacks to sub-exponential [Pie12]. Although there are no known reductions to standard worst-case lattice problems (e.g. GapSVP), recent work has been done to create reductions to similar worst-case problems such as binary SVP, used in the construction of collision-resistant hash functions [YZW19].

LPN has parallels with the more popular learning with errors (LWE) problem, with key differences being the modulus and the noise type. The binary modulus allows LPN-based systems to be much more efficient, since the ring operations in $\mathbb{Z}_2$ can be implemented with binary AND/XOR gates. From a theoretical perspective, this also makes LPN-based circuits have low depth. However, it is harder to find constructions with a binary modulus.

The main question that we survey in this paper is the following: under which noise regimes are LPN-based cryptographic constructions known? Two of the main noise regimes that are often studied are constant-noise and low-noise, for which we explore some examples.

2 Preliminaries

2.1 Notation

Column vectors are given by lowercase bold letters, e.g. $\mathbf{s}$. Row vectors are given by transposes of column vectors, e.g. $\mathbf{s}^T$. $\mathbb{Z}_n$ is the ring of integers modulo n. $\mathcal{B}_\mu$ is the Bernoulli distribution on $\mathbb{Z}_2$, taking the value 0 with probability μ and 1 with probability $1-\mu$. In constructions, λ denotes the security parameter.

The symbols $\leftarrow$ and $\xleftarrow{R}$ refer to sampling from a specified distribution or (resp.) uniformly at random. The symbols $\overset{C}{\approx}$ resp. $\overset{S}{\approx}$ indicate computational resp. statistical indistinguishability.

2.2 Hardness Assumptions

We define decisional LPN below. It is known that decisional LPN and computational LPN are polynomial equivalent [Blu94]. We will only use decisional LPN, and refer to it from now on as just LPN.

Definition 2.1. Let n be the hardness parameter, and let $\mu = \mu(n) \in (0, 1/2)$ be the noise rate, $m = m(n)$, $T = T(n)$. The $\mathsf{LPN}_{n,m,\mu}$ problem is said to be T-hard if for all probabilistic distinguishers $\mathcal{D}$ of runtime T, then for sufficiently large n

$$\left|\Pr\left[\mathcal{D}\left(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T\right)\right] - \Pr\left[\mathcal{D}(\mathbf{A}, \mathbf{u})\right]\right| \leq 1/T,$$

where

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_2^{n\times m}, \quad \mathbf{s} \xleftarrow{R} \mathbb{Z}_2^n, \quad \mathbf{e} \leftarrow \mathcal{B}_\mu^m, \quad \mathbf{u} \xleftarrow{R} \mathbb{Z}_2^m.$$

When $T = \mathsf{poly}(n)$ we often omit T.

As mentioned in the introduction, there are two main variants of LPN. When $m = \mathsf{poly}(n)$ and μ is constant, we call the assumption constant-noise LPN. When $m = O(n)$ and $\mu = O\left(n^{-1/2}\right)$, we refer to the assumption as low-noise LPN. Low-noise is generally considered a stronger assumption [DMN12, YZW19]: for example, it is known to give public-key cryptography, unlike constant noise.

There are a few variants that are also useful to state. The first is where we take the secret $\mathbf{s}$ to be Bernoulli, i.e.

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_2^{n\times m}, \quad \mathbf{s} \leftarrow \mathcal{B}_\mu^n, \quad \mathbf{e} \leftarrow \mathcal{B}_\mu^m, \quad \mathbf{u} \xleftarrow{R} \mathbb{Z}_2^m$$

$$\implies \left(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T\right) \overset{C}{\approx} (\mathbf{A}, \mathbf{u}).$$

in the above definition. Hardness of this variant follows from standard LPN. This turns out to be very useful in the low-noise setting. We also would like to support a matrix version where we take ℓ instances simultaneously:

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_2^{n\times m}, \quad \mathbf{S} \leftarrow \mathcal{B}_\mu^{\ell\times n}, \quad \mathbf{E} \leftarrow \mathcal{B}_\mu^{\ell\times m}, \quad \mathbf{U} \xleftarrow{R} \mathbb{Z}_2^{\ell\times m}.$$

$$\implies (\mathbf{A}, \mathbf{SA} + \mathbf{E}) \overset{C}{\approx} (\mathbf{A}, \mathbf{U}).$$

Hardness of the matrix version is immediate by a hybrid argument down the rows of $\mathbf{U}$.

3 Authentication Protocols from Constant-Noise LPN

Recall that in a secret-key authentication protocol, a prover $\mathcal{P}$ attempts to prove their identity to a verifier $\mathcal{V}$, where both parties have knowledge of some shared secret. There are many constructions of authentication protocols using LPN, stemming from the original HB protocol given by [HB001]. We will focus on HB and HB+, which illustrate simple applications of constant-noise LPN.

3.1 Authentication against Passive Adversaries

[HB001] gives a constant-noise LPN-based construction for a secret-key identification protocol secure in a passive attack setting, known as HB:

- Setup$\left(1^\lambda\right)$: Sample the shared secret, $\mathbf{z} \leftarrow \{0,1\}^n$.
- Verify: The verification procedure consists of m rounds. On round i,
  1. $\mathcal{V}$ samples a random challenge $\mathbf{c}_i \xleftarrow{R} \{0,1\}^n$, and sends it to $\mathcal{P}$.
  2. $\mathcal{P}$ computes $\mathbf{r}_i \leftarrow \mathbf{c}_i^T\mathbf{z}$. With probability μ, $\mathcal{P}$ sends $\mathbf{r}_i + 1$; otherwise $\mathcal{P}$ sends $\mathbf{r}_i$.
  3. $\mathcal{V}$ checks if $\mathbf{r}_i = \mathbf{c}_i^T\mathbf{z}$. If so, $\mathcal{P}$ passes the round.

  Finally, $\mathcal{V}$ accepts if $\mathcal{P}$ passes at least $(1-\mu)m$ of the rounds.

Correctness holds with constant probability. We can view the error as a binomial random variable X with parameters (m,μ). [Doe18] shows that an honest $\mathcal{P}$ will succeed with probability at least $1/4$ (since we want $\mu \geq 1/m$ so that noise is introduced). While this correctness guarantee is not ideal, it is much higher than a randomly guessing adversary (which wins with negligible probability). A more concrete analysis can be found in [HB001].

To do better than randomly guessing, a passive adversary would need some insight into $\mathbf{z}$. However, the transcripts a passive adversary sees form an LPN instance; Let $\mathbf{A}$ be a matrix whose rows are $\mathbf{c}_i^T$ and $\mathbf{r}$ be a vector whose entries are $\mathbf{r}_i$. Also, let $\mathbf{e}$ be a vector where $\mathbf{e}_i = 1$ if $\mathcal{P}$ introduces an error on round i, and 0 otherwise. $\mathbf{A}$ and $\mathbf{r}$ are the challenges and responses (resp.) the adverary sees, but under $\mathsf{LPN}_{n,m,\mu}$

$$(\mathbf{A},\mathbf{r}) \equiv \left(\mathbf{A},\mathbf{z}^T\mathbf{A}+\mathbf{e}^T\right) \overset{C}{\approx} (\mathbf{A},\mathbf{u})$$

where $\mathbf{u}$ is a random vector. Thus, an adversary cannot hope to do much better than random guessing.

Notably, an active adversary can break this scheme easily; by querying the same c_i multiple times, the majority output bit is correct with high probability, effectively removing the noise. With enough unique values of $\mathbf{c}_i$ one can solve for $\mathbf{z}$ directly via Gaussian elimination.

3.2 Authentication against Active Adversaries

[JW005] extends the above protocol to HB+, which provides security against active adversaries. Recall that an active adversary is one that is allowed to interact with $\mathcal{P}$ before attempting to authenticate with $\mathcal{V}$.

- Setup(1^λ): Sample two random bitstrings $\mathbf{x},\mathbf{y} \leftarrow \{0,1\}^n$. $\mathbf{x}$ and $\mathbf{y}$ together are the shared secret.
- Verify: The verification procedure consists of m rounds. On round i,
  1. $\mathcal{P}$ generates a random blinding vector $\mathbf{b}_i \xleftarrow{R} \{0,1\}^n$ and sends it to $\mathcal{V}$.
  2. $\mathcal{V}$ samples a random challenge vector $\mathbf{c}_i \xleftarrow{R} \{0,1\}^n$, and sends it to $\mathcal{P}$.
  3. $\mathcal{P}$ computes $\mathbf{r}_i \leftarrow \mathbf{b}_i^T\mathbf{x} + \mathbf{c}_i^T\mathbf{y}$. With probability μ, $\mathcal{P}$ sends $\mathbf{r}_i + 1$, otherwise $\mathcal{P}$ sends $\mathbf{r}_i$.
  4. $\mathcal{V}$ checks if $\mathbf{r}_i = \mathbf{b}_i^T\mathbf{x} + \mathbf{c}_i^T\mathbf{y}$. If so, $\mathcal{P}$ passes the round.

  Finally, $\mathcal{V}$ accepts if $\mathcal{P}$ passes at least $(1-\mu)m$ of the rounds.

This modification takes on a form similar to Σ-protocols, where now $\mathcal{P}$ also provides randomness to hide the shared secret. The adversary does not control $\mathbf{b}_i$, so $\mathbf{b}_i^T\mathbf{x} + \mathbf{e}_i$ appears random via LPN, entirely obscuring $\mathbf{c}_i^T\mathbf{y}$. Thus, an active adversary gains no advantage over a passive one in regular HB, so correctness and soundness hold as before. The full analysis can be found in [JW005].

4 Public Key Cryptography from Low-Noise LPN

There are several low-noise LPN-based constructions of public key cryptography in the literature [Ale03, DMN12, KMP14]. We will focus on [DMN12]. Our starting point is an IND-CPA public key scheme reminiscent of the LWE-based dual-Regev scheme. The role of rounding is replaced by efficient decoding algorithms for error-correcting codes; a soundness issue that arises uniquely in the LPN setting is solved with another error-correcting code. The base scheme can be combined with symmetric key encryption to yield a witness-recovering IND-CPA scheme (where decryption recovers the encryption randomness). Leveraging the properties of a high-distance code, we can further obtain an IND-CCA1 scheme by using the fact that codewords share few entries, allowing us to implement the decryption oracle for all but the challenge ciphertext. Full IND-CCA2 is a (nearly) black-box transformation. (Recall that in IND-CCA1, the adversary only has access to the decryption oracle before receiving the challenge; in IND-CCA2, the adversary always has access.)

4.1 An Impractical But Useful Example

To begin, we shall look to the dual-Regev scheme from LWE.

Construction 4.1.

- $\mathsf{Setup}\left(1^\lambda\right)$: Sample and compute

$$\begin{aligned}\mathbf{A} &\xleftarrow{\mathrm{R}} \mathbb{Z}_2^{n\times m},\\ \mathbf{s} &\leftarrow \mathcal{B}_\mu^n,\\ \mathbf{e} &\leftarrow \mathcal{B}_\mu^m,\\ \mathbf{b}^T &\leftarrow \mathbf{s}^T\mathbf{A}+\mathbf{e}^T.\end{aligned}$$

  The public key is $\mathsf{pk} = (\mathbf{A}, \mathbf{b})$. The secret key is $\mathsf{sk} = \mathbf{s}$.

- $\mathsf{Encrypt}\left(\mathsf{pk} = (\mathbf{A}, \mathbf{b}), m \in \mathbb{Z}_2\right)$: Sample and compute

$$\begin{aligned}\mathbf{r} &\leftarrow \mathcal{B}_\mu^m,\\ \mathbf{e}_1 &\leftarrow \mathcal{B}_\mu^n,\\ e_2 &\leftarrow \mathcal{B}_\mu.\end{aligned}$$

  The ciphertext is $\mathsf{ct} = (\mathbf{A}\mathbf{r}+\mathbf{e}_1, \mathbf{b}^T\mathbf{r}+e_2+m)$.

- $\mathsf{Decrypt}\left(\mathsf{sk} = \mathbf{s}, \mathsf{ct} = (\mathbf{c}_1, c_2)\right)$: Compute and output $c_2 - \mathbf{s}^T\mathbf{c}_1$.

To analyze correctness, we can compute algebraically that

$$c_2 - \mathbf{s}^T\mathbf{c}_1 = m + \mathbf{e}^T\mathbf{r} + \mathbf{e}_2 - \mathbf{s}^T\mathbf{e}_1.$$

For this to equal m, we need the noise term $\mathbf{e}^T\mathbf{r}+\mathbf{e}_2-\mathbf{s}^T\mathbf{e}_1$ to be zero. Using the low-noise LPN assumption, the key observation is that we can union bound the probability that (e.g.) $\mathbf{e}^T\mathbf{r} \neq 0$:

$$\begin{aligned}\Pr\left[\mathbf{e}^T\mathbf{r} \neq 0\right] &\leq \Pr\left[\exists i : \mathbf{e}_i = \mathbf{r}_i = 1\right]\\ &= m\mu^2\\ &= O(1).\end{aligned}$$

With the appropriate choices of parameters, we can get correctness bounded above $1/2$. The above analysis also reveals the motivation behind using the low-noise domain.

Of course, we would like better correctness guarantees. A clever insight is to use an error correcting code combined with parallel instances of the above scheme: encryption followed by decryption acts like a noisy channel, which is then corrected by the code.

4.2 Witness-Recovering IND-CPA

With the necessary modifications, we are ready to see the full (witness-recovering) IND-CPA scheme.

Construction 4.2. Let $n = n(\lambda)$, $\ell_1, \ell_2, \ell_3 = O(n)$ and $\mu = O\left(n^{-1/2}\right)$. Let $\mathcal{C}$ be a binary *linear* $[\ell_2, n]$ code with efficient encoding and decoding $(\mathsf{Encode}_{\mathcal{C}}, \mathsf{Decode}_{\mathcal{C}})$. Let $\mathcal{D}$ be a binary (ℓ_3, n) code with efficient encoding and decoding $(\mathsf{Encode}_{\mathcal{D}}, \mathsf{Decode}_{\mathcal{D}})$.

- $\mathsf{Setup}\left(1^{\lambda}\right)$: Sample and compute

$$\begin{aligned}
\mathbf{A} &\xleftarrow{R} \mathbb{Z}_2^{\ell_1 \times n} \\
\mathbf{S} &\leftarrow \mathcal{B}_{\mu}^{\ell_2 \times \ell_1} \\
\mathbf{E} &\leftarrow \mathcal{B}_{\mu}^{\ell_2 \times n} \\
\mathbf{B} &\leftarrow \mathbf{SA} + \mathbf{E} \\
\mathbf{C} &\xleftarrow{R} \mathbb{Z}_2^{\ell_3 \times n}
\end{aligned}$$

  Output the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{B}, \mathbf{C})$ and the secret key $\mathsf{sk} = \mathbf{S}$.

- $\mathsf{Encrypt}\left(\mathsf{pk} = (\mathbf{A}, \mathbf{B}, \mathbf{C}), \mathbf{m} \in \mathbb{Z}_2^n\right)$: Sample

$$\begin{aligned}
\mathbf{r} &\leftarrow \mathcal{B}_{\mu}^{n} \\
\mathbf{e}_1 &\leftarrow \mathcal{B}_{\mu}^{\ell_1} \\
\mathbf{e}_2 &\leftarrow \mathcal{B}_{\mu}^{\ell_2} \\
\mathbf{e}_3 &\leftarrow \mathcal{B}_{\mu}^{\ell_3}
\end{aligned}$$

  Output the ciphertext

$$\mathsf{ct} = (\mathbf{Ar} + \mathbf{e}_1, \mathbf{Br} + \mathbf{e}_2 + \mathsf{Encode}_{\mathcal{C}}(r), \mathbf{Cr} + \mathbf{e}_3 + \mathsf{Encode}_{\mathcal{D}}(m)).$$

- $\mathsf{Decrypt}\left(\mathsf{sk} = \mathbf{S}, \mathsf{ct} = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)\right)$: Compute the encryption randomness

$$r \leftarrow \mathsf{Decode}_{\mathcal{C}}\left(\mathbf{c}_2 - \mathbf{Sc}_1\right).$$

  Then compute and output the message

$$\mathsf{Decode}_{\mathcal{D}}\left(\mathbf{c}_3 - \mathbf{Cr}\right).$$

  If either decoding step fails, output $\perp$.

Theorem 4.3. *The above construction is correct (for sufficiently large λ).*

Proof. We can verify algebraically that

$$\mathbf{c}_2 - \mathbf{S}\mathbf{c}_1 = \mathsf{Encode}_{\mathbf{C}}(\mathbf{r}) + \mathbf{E}\mathbf{r} + \mathbf{e}_2 - \mathbf{S}\mathbf{e}_1$$

$\mathsf{Decode}_{\mathcal{C}}$ will successfully recover $\mathbf{r}$ if the error term $\mathbf{E}\mathbf{r} + \mathbf{e}_2 - \mathbf{S}\mathbf{e}_1$ is small enough (and intuitively it is small, as each individual variable is small). Similarly,

$$\mathbf{c}_3 - \mathbf{C}\mathbf{r} = \mathsf{Encode}_{\mathcal{D}}(\mathbf{m}) + \mathbf{e}_3$$

will recover $\mathbf{m}$ when decoded if the error $\mathbf{e}_3$ is small enough. The precise details of the error analysis (and parameters) are deferred to [DMN12], but it generally models the ideas from the original impractical scheme. □

Theorem 4.4. *The above construction is IND-CPA secure.*

Proof. Consider the sequence of hybrid games:

- Hyb_0: The normal IND-CPA game.
- Hyb_1: Like Hyb_0, but instead of $\mathbf{B}$, $\mathbf{B}^*$ is chosen uniformly at random during Setup.
- Hyb_2: Like Hyb_1, but instead of $\mathsf{ct} = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$, each component in the challenge ciphertext $\mathsf{ct}^* = (\mathbf{c}_1^*, \mathbf{c}_2^*, \mathbf{c}_3^*)$ is chosen uniformly at random.

Obviously no adversary can win Hyb_2. Observe that $\mathsf{Hyb}_0 \overset{\mathrm{C}}{\approx} \mathsf{Hyb}_1$ since under low-noise LPN, the views satisfy

$$(\mathbf{A}, \mathbf{B}) \equiv (\mathbf{A}, \mathbf{S}\mathbf{A} + \mathbf{E}) \overset{\mathrm{C}}{\approx} (\mathbf{A}, \mathbf{B}^*).$$

Similarly, $\mathsf{Hyb}_1 \overset{\mathrm{C}}{\approx} \mathsf{Hyb}_2$ by three more applications of LPN. Obviously

$$(\mathbf{A}, \mathbf{c}_1) \equiv (\mathbf{A}, \mathbf{A}\mathbf{r} + \mathbf{e}_1) \overset{\mathrm{C}}{\approx} \left(\mathbf{A}, \mathbf{c}_1^*\right)$$

The second component is a bit more tricky since we are encrypting the randomness itself. However, because $\mathcal{C}$ is linear, it has a generator matrix $\mathbf{G}$. Thus we can factor as follows:

$$(\mathbf{B}, \mathbf{c}_2) \equiv (\mathbf{B}, \mathbf{B}\mathbf{r} + \mathbf{e}_2 + \mathsf{Encode}_{\mathcal{C}}(\mathbf{r})) \equiv (\mathbf{B}, (\mathbf{B} + \mathbf{G})\mathbf{r} + \mathbf{e}_2) \overset{\mathrm{C}}{\approx} \left(\mathbf{B} - \mathbf{G}, \mathbf{c}_2^*\right) \equiv \left(\mathbf{B}, \mathbf{c}_2^*\right).$$

The final component is easier, since there is no dependence:

$$(\mathbf{C}, \mathbf{c}_3) \equiv (\mathbf{C}, \mathbf{C}\mathbf{r} + \mathbf{e}_3 + \mathsf{Encode}_{\mathbf{D}}(\mathbf{m})) \overset{\mathrm{C}}{\approx} \left(\mathbf{C}, \mathbf{c}_3^* + \mathsf{Encode}_{\mathbf{D}}(\mathbf{m})\right) \overset{\mathrm{C}}{\approx} \left(\mathbf{C}, \mathbf{c}_3^*\right).$$

□

In the above construction, we use $\mathbf{A}, \mathbf{B}$ as a normal public key encryption scheme (with the linearity trick that lets us encrypt the randomness), and $\mathbf{C}$ as a symmetric key scheme. This gives the desired witness-recovery property.

4.3 IND-CCA1 and IND-CCA2

The IND-CPA construction can be extended to IND-CCA1 by leveraging the witness-recovery property [DMN12]: decryption shall reject any ciphertexts with large randomness or error. Now, let

$$\mathbf{y} = \mathbf{c}_2 - \mathbf{S}\mathbf{c}_1 = \mathsf{Encode}_{\mathcal{C}}(\mathbf{r}) + \mathbf{E}\mathbf{r} + \mathbf{e}_2 - \mathbf{S}\mathbf{e}_1,$$

i.e. the value passed to $\mathsf{Decode}_{\mathcal{C}}$ in decryption. A crucial observation is that if we only know the ith row of $\mathbf{S}$, then we can compute the ith component $\mathbf{y}_i$ of $\mathbf{y}$. If we know most of the rows of $\mathbf{S}$, then we can feed known values of $\mathbf{y}_i$ with erasures in unknown positions into an error/erasure decoding algorithm for $\mathcal{C}$ to recover $\mathbf{r}$. This observation will turn out to be crucial in implementing the decryption oracle.

We can now give a high level overview of the scheme. In Setup, we generate q different matrices $\mathbf{B}_1, \ldots, \mathbf{B}_q$ and $\mathbf{T}_1, \ldots, \mathbf{T}_q$. For each ciphertext, we now generate and embed a random tag τ_0 into the ciphertext. τ_0 is encoded with a q-ary code $\mathcal{E}$ of high distance, obtaining τ. Then τ is used to assemble the rows of $\mathbf{B}_i$ and $\mathbf{T}_i$ into an derived instance key pair for encryption and decryption (the jth row of the derived keys $\mathbf{B}_\tau$ and $\mathbf{T}_\tau$ are given by the jth row of $\mathbf{B}_{\tau_j}$ resp. $\mathbf{T}_{\tau_j}$), used in the same manner as the IND-CPA construction.

The key element of the security proof is that upon choosing the tag for the challenge ciphertext τ^*, a decryption attempt with a different tag $\tau \neq \tau^*$ will have a derived key pair sharing few rows with the derived key from τ^*; this is due to the high distance of $\mathcal{E}$. Combined with the initial observation of this section, we can set the relatively few rows that are shared between the derived secret keys with erasures to entirely compute the simulated decryption. The explicit construction and security proof is left to [DMN12].

The IND-CCA2 construction is less novel and is nearly a black box transformation. We need only replace the random tags τ_0 with verification keys from a one-time signature scheme.

5 Conclusion and Related Work

Using various noise levels of LPN, we are able to construct a wide variety of cryptographic primitives. [Pie12] provides a construction of PRGs and OWFs from LPN, while [GRS08] gives a simple symmetric key scheme with adaptive IND-CPA. [JKP12] gives a very simple construction of statistically binding and computationally hiding commitments, as well as zero-knowledge proofs. Somewhat recently, [YZW19] solved an open problem by constructing collision-resistant hash functions from LPN, albeit with lower noise ($\mu = \log^2 n/n$).

However, it appears as though LPN is a strictly weaker construct than LWE. Nearly all primitives from LPN can also be created from LWE, but the converse is not true. Crucially, a result by [Bra13] shows that current approaches to LPN-based encryption, namely low-noise constructions analogous to LWE encryption schemes, *cannot* achieve fully homomorphic encryption. This result does not rule out FHE from LPN, but rather implies that a new paradigm for LPN encryption is necessary to achieve it.

It appears as though LPN's greatest strength lies in its efficiency. As LPN is computed over $\mathbb{Z}_2$, addition and multiplication can be computed via XOR/AND gates. Similar to RingLWE, a variant of LWE which leverages polynomial rings to improve performance, an LPN counterpart also exists: RingLPN. While not as efficient as standard cryptosystems in use today, RingLPN has shown promising results in performance. [HKL12] provides an authentication scheme based on RingLPN, which performs only twice as slowly as AES-based authentication when allowed some precomputation. [YGZ18] provides an efficient encryption scheme using RingLPN, which decrypts faster than RSA. It is worth noting that no research has been done to directly compare the performance of LPN schemes to their LWE counterparts.

A final aspect to comment on is in the interplay between LPN and LWE. LPN constructions are generally coding-theoretic in nature, and coding theory has been extensively studied—there are bound to be more connections to be made. Furthermore, LPN constructions often parallel LWE constructions. For example, the IND-CCA public-key scheme discussed is based similar to an LWE-based scheme in which short dual-basis lattices with decoding algorithms are used, which as the authors of [DMN12] note are essentially a Euclidean analogue of standard error-correcting codes. Thus, developments in LWE can influence LPN, but conversely developments in LPN and coding theory in general can also motivate LWE constructions.

References

[Ale03] Michael Alekhnovich. More on average case vs approximation complexity. 2003.

[Blu94] Cryptographic primitives based on hard learning problems. 1994.

[Bra13] Zvika Brakerski. When homomorphism becomes a liability. 2013.

[DMN12] IND-CCA secure cryptography based on a variant of the LPN problem. 2012.

[Doe18] Benjamin Doerr. An elementary analysis of the probability that a binomial random variable exceeds its expectation. 2018.

[GRS08] How to encrypt with the LPN problem. 2008.

[HB001] Secure human identification protocols. 2001.

[HKL12] Lapin: An efficient authentication protocol based on ring-LPN. 2012.

[JKP12] Commitments and efficient zero-knowledge proofs from learning parity with noise. 2012.

[JW005] Authenticating pervasive devices with human protocols. 2005.

[KMP14] Simple chosen-ciphertext security from low-noise LPN. 2014.

[Pie12] Krzysztof Pietrzak. Cryptography from learning parity with noise. 2012.

[YGZ18] A practical public key encryption scheme based on learning parity with noise. 2018.

[YZW19] Collision resistant hashing from sub-exponential learning parity with noise. 2019.