# When Homomorphism Becomes a Liability

Zvika Brakerski*

Stanford University
zvika@stanford.edu

**Abstract.** We show that an encryption scheme cannot have a simple decryption function and be homomorphic at the same time, even with added noise. Specifically, if a scheme can homomorphically evaluate the majority function, then its decryption cannot be weakly-learnable (in particular, linear), even if the probability of decryption error is high. (In contrast, without homomorphism, such schemes do exist and are presumed secure, e.g. based on LPN.)
An immediate corollary is that known schemes that are based on the hardness of decoding in the presence of *low hamming-weight noise* cannot be fully homomorphic. This applies to known schemes such as LPN-based symmetric or public key encryption.
Using these techniques, we show that the recent candidate fully homomorphic encryption, suggested by Bogdanov and Lee (ePrint '11, henceforth BL), is insecure. In fact, we show two attacks on the BL scheme: One that uses homomorphism, and another that directly attacks a component of the scheme.

## 1 Introduction

An encryption scheme is called homomorphic if there is an efficient transformation that given $\mathsf{Enc}(m)$ for some message $m$, and a function $f$, produces $\mathsf{Enc}(f(m))$ using only public information. A scheme that is homomorphic w.r.t all efficient $f$ is called fully homomorphic (FHE). Homomorphic encryption is a useful tool in both theory and practice and is extensively researched in recent years (see [20] for survey), and a few candidates for full homomorphism are known.

Most of these candidates [9, 10, 19, 6, 7, 11, 5, 12, 4] are based (either explicitly or implicitly) on *lattice assumptions* (the hardness of approximating short vectors in certain lattices). In particular, the learning with errors (LWE) assumption proved to be very useful in the design of such schemes. The one notable exception is [22], but even that could be thought of as working over an appropriately defined lattice over the integers.

An important open problem is, therefore, to diversify and base fully homomorphic encryption on different assumptions (so as to not put all the eggs in one basket). One appealing direction is to try to use the learning parity with noise (LPN) problem, which is very similar in syntax to LWE: Making a vast

* Supported by a Simons Postdoctoral Fellowship and by DARPA.

generalization, LWE can be interpreted as a decoding problem for a linear code, where the noise comes from a family of *low norm* vectors. Namely, each coordinate in the code suffers from noise, but this noise is relatively small (this requires that the code is defined over a large alphabet). The LPN assumption works over the binary alphabet and requires that the noise has low hamming weight, namely that only a small number of coordinates are noisy, but in these coordinates the noise amplitude can be large. While similar in syntax, a direct connection between these two types of assumptions is not known.

While an LPN-based construction is not known, recently Bogdanov and Lee [3] presented a candidate, denoted by BL throughout this manuscript, that is based on a different low hamming-weight decoding problem: They consider a carefully crafted code over a large alphabet and assume that decoding in the presence of low-hamming-weight noise is hard.

In this work, we show that not only that BL's construction is insecure, but rather the entire approach of constructing code based homomorphic encryption analogously to the LWE construction cannot work. We stress that we don't show that FHE cannot be based on LPN (or other code based assumptions), but rather that the decryption algorithm of such scheme cannot take the naïve form. (In particular this applies to the attempt to add homomorphism to schemes such as [1, 13, 2].)

### 1.1 Our Results

Our main result shows that encryption schemes with learnable decryption functions cannot be homomorphic, even if a high probability of decryption error is allowed. In particular, such schemes cannot evaluate the majority function. This extends the result of Kearns and Valiant [15] (slightly extended by Klivans and Sherstov [16]) that learnability breaks security for schemes with negligible decryption error. In other words, homomorphic capabilities can sometimes make noisy learning become no harder than noiseless learning.

We use a simplified notion of learning, which essentially requires that given polynomially many labeled samples (from an arbitrary distribution), the learner's hypothesis correctly computes the label for the next sample with probability, say, 0.9. We show that this notion, that we call *sc-learning*, is equivalent to *weak learning* defined in [15]. This allows us to prove the following theorem (in Section 3).

**Theorem A.** *An encryption scheme whose decryption function is sc- or weakly-learnable, and whose decryption error is* $1/2 - 1/\mathrm{poly}(n)$*, cannot homomorphically evaluate the majority function.*

Since it is straightforward to show that linear functions are learnable (as well as, e.g., low degree polynomials), the theorem applies to known LPN based schemes such as [1, 13, 2]. This may not seem obvious at first: The decryption circuit of the aforementioned schemes is (commonly assumed to be) hard to learn, and their decryption error is negligible, so they seem to be out of the scope of

our theorem. However, looking more closely, the decryption circuits consist of an inner product computation with the secret key, followed by additional post-processing. One can verify that if the post processing is not performed, then correct decryption is still achieved with probability $> 1/2+1/\text{poly}$. Thus we can apply our theorem and rule out majority-homomorphism.

Similar logic rules out the homomorphism of the BL candidate-FHE. While Theorem A does not apply directly (since the decryption of BL is not learnable out of the box), we show that it contains a sub-scheme which is linear (and thus learnable) and has sufficient homomorphic properties to render it insecure.

**Theorem B.** *There is a successful polynomial time CPA attack on the BL scheme.*

We further present a different attack on the BL scheme, targeting one of its building blocks. This allows us to not only distinguish between two messages like the successful CPA attack above, but rather decrypt any ciphertext with probability $1 - o(1)$.

**Theorem C.** *There is a polynomial time algorithm that decrypts the BL scheme.*

The BL scheme and the two breaking algorithms are presented in Section 4.

### 1.2 Our Techniques

Consider a simplified case of Theorem A, where the scheme's decryption function is learnable given $t$ labeled samples, and the decryption error is (say) $1/(10(t+1))$. The proof in this case is straightforward: Generate $t$ labeled samples by just encrypting random messages, and feed them to the learner. Then use the learner's output hypothesis to decrypt the challenge ciphertext. We can only fail if either the learner fails (which happens with probability 0.1) or if one of the samples we draw (including the challenge) are not correctly decryptable, in which case our labeling is wrong and therefore the learner provides no guarantee (which again happens with at most 0.1 probability). The union bound implies that we can decrypt a random ciphertext with probability 0.8, which immediately breaks the scheme. Note that we did not use the homomorphism of the scheme at all, indeed this simplified version is universally true even without assuming homomorphism, and is very similar to the arguments in [15, 16]. (Some subtleties arise since we allow a non-negligible fraction of "dysfunctional" keys that induce a much higher error rate than others.)

The next step is to allow decryption error $1/2 - \epsilon$, which requires use of homomorphism. The idea is to use the homomorphism in order to reduce the decryption error and get back to the previous case (in other words, reducing the noise in a noisy learning problem). Consider a scheme that encrypts a message by generating many encryptions (say $k$) of that message, and then applying homomorphic majority on those ciphertexts and outputting the result. The security of this scheme directly reduces from that of the original scheme, and it has the

same decryption function. However, now the decryption error drops exponentially with $k$. This is because in order to get an error in the new scheme, at least $k/2$ out of the $k$ encryptions need to have errors. Since the expected number is $(1/2 - \epsilon)k$, the Chernoff bound implies the result by choosing $k$ appropriately.

To derive Theorem B, we need to show that linear functions are learnable:[1] Assume that the decryption function is an inner product between the ciphertext and the secret key (both being $n$-dimensional vectors over a field $\mathbb{F}$). We will learn these functions by taking $O(n)$ labeled samples. Then, given the challenge, we will try to represent it as a linear combination of the samples we have. If we succeed, then the appropriate linear combination of the labels will be the value of the function on the challenge. We show that this process fails only with small constant probability (intuitively, since we take $O(n)$ sample vectors from a space of dimension at most $n$).

We then show that BL uses a sub-structure that is both linearly decryptable and allows for homomorphism of (some sort of) majority. Theorem B thus follows similarly to Theorem A.

For Theorem C, we need to dive into the guts of the BL scheme. We notice that BL use homomorphic majority evaluation in one of the lower abstraction levels of their scheme. This allows us to break this abstraction level using only linear algebra (in a sense, the homomorphic evaluation is already "built in"). A complete break of BL follows.

### 1.3 Other Related Work

An independent work by Gauthier, Otmani and Tillich [8] shows an interesting direct attack on BL's hardness assumption (we refer to it as the "GOT attack"). Their attack is very different from ours and takes advantage of the resemblance of BL's codes and Reed-Solomon codes as we explain below.

BL's construction relies on a special type of error correcting code. Essentially, they start with a Reed-Solomon code, and replace a small fraction of the rows of the generating matrix with a special structure. The homomorphic properties are only due to this small fraction of "significant" rows, and the secret key is chosen so as to nullify the effect of the other rows.

The GOT attack uses the fact that under some transformation (componentwise multiplication), the dimension of Reed-Solomon codes can grow by at most a factor of two. However, if a code contains "significant" rows, then the dimension can grow further. This allows to measure the number of significant rows in a given code. One can thus identify the significant rows by trying to remove one row at a time from the code and checking if the dimension drops. If yes then that row is significant. Once all significant rows have been identified, the secret key can be retrieved in a straightforward manner.

However, it is fairly easy to immunize BL's scheme against the GOT attack. As we explained above, the neutral rows do not change the properties of the

[1] We believe this was known before, but since we could not find an appropriate reference, we provide a proof.

encryption scheme, so they may as well be replaced by random rows. Since the dimension of random codes grows very rapidly under the GOT transformation, their attack will not work in such case.

Our attack, on the other hand, relies on certain functional properties that BL use to make their scheme homomorphic. Thus a change in the scheme that preserves these homomorphic properties cannot help to overcome our attack. In light of our attack, it is interesting to investigate whether the GOT attack can be extended to the more general case.

## 2 Preliminaries

We denote scalars using plain lowercase ($x$), vectors using bold lowercase ($\mathbf{x}$ for column vector, $\mathbf{x}^T$ for row vector), and matrices using bold uppercase ($\mathbf{X}$). We let $\mathbf{1}$ denote the all-one vector (the dimension will be clear from the context). We let $\mathbb{F}_q$ denote a finite field of cardinality $q \in \mathbb{N}$, with efficient operations (we usually don't care about any other property of the field).

### 2.1 Properties of Encryption Schemes

A public key encryption scheme is a tuple of algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, such that: $\mathsf{Gen}(1^n)$ is the key generation algorithm that produces a pair of public and secret keys $(pk, sk)$; $\mathsf{Enc}_{pk}(m)$ is a randomized encryption function that takes a message $m$ and produces a ciphertext. In the context of this work, messages will only come from some predefined field $\mathbb{F}$; $\mathsf{Dec}_{sk}(c)$ is the decryption function that decrypts a ciphertext $c$ and produces the message. Optimally, $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\cdot))$ is the identity function, but in some schemes there are decryption errors.

The probability of decryption error is taken over the randomness used to generate the keys for the scheme, and over the randomness used in the encryption function (we assume the decryption is deterministic). Since in our case the error rates are high (approaching 1/2), the effect of bad keys is different from that of bad encryption randomness, and we thus measure the two separately. We allow a small fraction of the keys (one percent, for the sake of convenience) to have arbitrarily large decryption error, and define the decryption error $\epsilon$ to be the maximal error over the 99% best keys. While the constant 1% is arbitrary and chosen so as to not over-clutter notation, we will discuss after presenting our results how they generalize to other values. The formal definition follows.

**Definition 2.1.** *An encryption scheme is said to have decryption error $< \epsilon$ if with probability at least 0.99 over the key generation it holds that*

$$\max_m \{\Pr[\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m)) \neq m]\} < \epsilon \ ,$$

*where the probability is taken over the random coins of the encryption function.*

We use the standard definition of security against chosen plaintext attacks (CPA): The attacker receives a public key and chooses two values $m_0, m_1$. The

attacker then receives a ciphertext $c = \mathsf{Enc}_{pk}(m_b)$, where $b \in \{0,1\}$ is a random bit that is unknown to the attacker. The attacker needs to decide on a guess $b' \in \{0,1\}$ as to the value of $b$. We say that the scheme is broken if there is a polynomial time attacker for which $\Pr[b' = b] \geq 1/2 + 1/\mathrm{poly}(n)$ (where $n$ is the security parameter). Recall that this notion is equivalent to the notion of semantic security [14].

In addition, we will say that a scheme is *completely broken* if there exists an adversary that upon receiving the public key and $\mathsf{Enc}_{pk}(m)$ for *arbitrary value of* $m$, returns $m$ with probability $1 - o(1)$.

While we discuss homomorphic properties of encryption schemes, we will only use homomorphism w.r.t the majority function. We define the notion of $k$-majority-homomorphism below.

**Definition 2.2.** *A public-key encryption scheme is* $k$-majority-homomorphic *(where $k$ is a function of the security parameter) if there exists a function* $\mathsf{MajEval}$ *such that with probability* 0.99 *over the key generation, for any sequence of ciphertexts output by* $\mathsf{Enc}_{pk}(\cdot)$*: $c_1, \ldots, c_k$, it holds that*

$$\mathsf{Dec}_{sk}(\mathsf{MajEval}_{pk}(c_1, \ldots, c_k)) = \mathsf{Majority}(\mathsf{Dec}_{sk}(c_1), \ldots, \mathsf{Dec}_{sk}(c_k)) \ .$$

Again we allow some "slackness" by allowing some of the keys to not abide the homomorphism.

We note that Definition 2.2 above is a fairly strong notion of homomorphism in two aspects: First, it requires that homomorphism holds even for ciphertexts with decryption error. Second, we do not allow $\mathsf{MajEval}$ to introduce error for "good" key pairs. Indeed, known homomorphic encryption schemes have these properties, but it is interesting to try to bypass our negative results by finding schemes that do not have them.

Schemes with linear decryption, as defined below, have a special role in our attack on BL.

**Definition 2.3.** *An encryption scheme is $n$-linearly decryptable if its secret key is of the form $sk = \mathbf{s} \in \mathbb{F}^n$, for some field $\mathbb{F}$, and its decryption function is*

$$\mathsf{Dec}_{sk}(\mathbf{c}) = \langle \mathbf{s}, \mathbf{c} \rangle \ .$$

### 2.2 Spanning Distributions over Low Dimensional Spaces

We will use a lemma that shows that any distribution over a low dimensional space is easy to span in the following sense: Given sufficiently many samples from the distribution (a little more than the dimension of the support), we are guaranteed that any new vector falls in the span of previous samples. This lemma will allow us to derive a (distribution-free) learner for linear functions (see Section 2.3).

We speculate that this lemma is already known, since it is fairly general and very robust to the definition of dimension (e.g. it also applies to non-linear spaces).

**Lemma 2.4.** *Let $\mathcal{S}$ be a distribution over a linear space $S$ of dimension $s$. For all $k$, define*

$$\delta_k \triangleq \Pr_{\mathbf{v}_1,\ldots,\mathbf{v}_k \xleftarrow{\$} \mathcal{S}} \left[\mathbf{v}_k \notin \mathrm{Span}\left\{\mathbf{v}_1,\ldots,\mathbf{v}_{k-1}\right\}\right] \ .$$

*Then $\delta_k \leq s/k$.*

*Proof.* Notice that by symmetry $\delta_i \geq \delta_{i+1}$ for all $i$. Let $D_i$ denote the (random variable) dimension of $\mathrm{Span}\left\{\mathbf{v}_1,\ldots,\mathbf{v}_i\right\}$. Note that always $D_i \leq s$.

Let $E_i$ denote the event $\mathbf{v}_i \notin \mathrm{Span}\left\{\mathbf{v}_1,\ldots,\mathbf{v}_{i-1}\right\}$ and let $\mathbb{1}_{E_i}$ denote the indicator random variable for this event. Then $\delta_i = \Pr[E_i] = \mathbb{E}[\mathbb{1}_{E_i}]$. By definition,

$$D_k = \sum_{i=1}^{k} \mathbb{1}_{E_i} \ .$$

Therefore

$$s \geq \mathbb{E}[D_k] = \mathbb{E}\left[\sum_{i=1}^{k} \mathbb{1}_{E_i}\right] = \sum_{i=1}^{k} \Pr[E_i] = \sum_{i=1}^{k} \delta_i \geq k \cdot \delta_k \ ,$$

and the lemma follows.

### 2.3 Learning

In this work we use two equivalent notions of learning: weak-learning as defined in [15], and an equivalent simplified notion that we call single-challenge-learning (sc-learning for short). The latter will be more convenient for our proofs, but we show that the two are equivalent. We will also show that linear functions are sc-learnable.

*Notions of Learning.* We start by introducing the notion of weak-learnability.

**Definition 2.5 (weak-learning [15]).** *Let $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ be an ensemble of binary functions. A weak learner for $\mathcal{F}$ with parameters $(t, \epsilon, \delta)$ is a polynomial time algorithm $A$ such that for any function $f \in \mathcal{F}_n$ and for any distribution $\mathcal{D}$ over the inputs to $f$, the following holds. Let $x_1,\ldots,x_{t+1} \xleftarrow{\$} \mathcal{D}$, and let $h$ ("the hypothesis") be the output of $A(1^n, (x_1, f(x_1)),\ldots,(x_t, f(x_t)))$. Then*

$$\Pr_{x_1,\ldots,x_t}\left[\Pr_{x_{t+1}}\left[h(x_{t+1}) \neq f(x_{t+1})\right] > \epsilon\right] \leq \delta \ .$$

*We say that $\mathcal{F}$ is weakly learnable if there exists a weak learner for $\mathcal{F}$ with parameters $t = \mathrm{poly}(n)$, $\epsilon \leq 1/2 - 1/\mathrm{poly}(n)$, $\delta \leq 1 - 1/\mathrm{poly}(n)$. (We also require that the output hypothesis $h$ is polynomial time computable.)*

We next define our notion of $(t, \eta)$-sc-learning, which essentially corresponds to the ability to launch a $t$-query CPA attack on an (errorless) encryption scheme, and succeed with probability $\eta$. (The initials "sc" stand for "single challenge", reflecting the fact that a CPA attacker only receives a single challenge ciphertext.)

**Definition 2.6 (sc-learning).** *Let $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ be an ensemble of functions. A $(t,\eta)$-sc-learner for $\mathcal{F}$ is a polynomial time algorithm $A$ such that for any function $f \in \mathcal{F}_n$ and for any distribution $\mathcal{D}$ over the inputs to $f$, the following holds. Let $x_1,\ldots,x_{t+1} \stackrel{\$}{\leftarrow} \mathcal{D}$, and let $h$ ("the hypothesis") be the output of $A(1^n,(x_1,f(x_1)),\ldots,(x_t,f(x_t)))$. Then $\Pr[h(x_{t+1}) \neq f(x_{t+1})] \leq \eta$, where the probability is taken over the entire experiment.*

*We say that $\mathcal{F}$ is $(t,\eta)$-sc-learnable if it has a polynomial time $(t,\eta)$-sc-learner for it. We say that a* binary *$\mathcal{F}$ is sc-learnable if $t = \mathrm{poly}(n)$ and $\eta \leq 1/2 - 1/\mathrm{poly}(n)$. (We also require that the output hypothesis $h$ is polynomial time computable.)*

Since sc-learning only involves one challenge, we do not define the "confidence" and "accuracy" parameters $(\delta, \epsilon)$ separately as in the definition of weak-learning.

We note that both definitions allow for *improper learning* (namely, the hypothesis $h$ does not need to "look like" an actual decryption function).

*Equivalence Between Notions.* The equivalence of the two notions is fairly straightforward. Applying boosting [18] shows that sc-learning, like weak-learning, can be amplified.

**Claim 1.** *If $\mathcal{F}$ is sc-learnable then it is weak-learnable.*

*Proof.* This follows by a Markov argument: Consider a $(t,\eta)$-sc-learner for $\mathcal{F}$ (recall that $\eta \leq 1/2 - 1/\mathrm{poly}(n)$) and let $\delta = 1 - 1/\mathrm{poly}(n)$ be such that $\eta/\delta \leq 1/2 - \mathrm{poly}(n)$ (such $\delta$ must exist). Then letting $\epsilon \triangleq \eta/\delta$ finishes the argument.

The opposite direction will give us very strong amplification of learning by applying boosting [18]. The boosting algorithm can amplify the $\epsilon, \delta$ values of a weak learner to arbitrarily small values, at the cost of increasing the number of required samples.

**Claim 2.** *If $\mathcal{F}$ is weak-learnable then it is $(\mathrm{poly}(n,1/\eta),\eta)$-sc-learnable for all $\eta$.*

*Proof.* Let $\mathcal{F}$ be weak-learnable. Then by boosting [18] it is also PAC learnable [21]. Namely there is a learner with parameters $(\mathrm{poly}(n,1/\epsilon,1/\delta),\epsilon,\delta)$ for any inversely polynomial $\epsilon,\delta$. Setting $\epsilon = \delta = \eta/2$, the claim follows.

*Learning Linear Functions.* The following corollary (of Lemma 2.4) shows a simple direct construction of an sc-learner for the class of linear functions.[2]

**Corollary 2.7.** *Let $\mathcal{F}_n$ be a class of $n$-dimensional linear functions over a field $\mathbb{F}$. Then $\mathcal{F} = \{\mathcal{F}_n\}_n$ is $(10n, 1/10)$-sc-learnable.*

[2] The learner works even when the function class is not binary, which is only an advantage. The binary case follows by considering distributions supported only over the pre-images of $0, 1$.

*Proof.* We note that for any linear function $f : \mathbb{F}^n \to \mathbb{F}$, the set $\{(\mathbf{x}, f(\mathbf{x}))\}_{\mathbf{x}\in\mathbb{F}^n}$ is an $n$-dimensional linear subspace of $\mathbb{F}^{n+1}$.

The learner $A$ works as follows. It is given $t = 10n$ samples $\mathbf{v}_i \triangleq (\mathbf{x}_i, f(\mathbf{x}_i)) \in \mathbb{F}^{n+1}$. Using Gaussian elimination, $A$ will find $\mathbf{s} \in \mathbb{F}^n$ such that $(-\mathbf{s}, 1) \in \mathrm{Ker}\{\mathbf{v}_i\}_{i\in[t]}$ (note that such must exist). Finally $A$ will output the hypothesis $h(\mathbf{x}) = \langle \mathbf{s}, \mathbf{x}\rangle$.

Correctness follows using Lemma 2.4. We let the distribution $\mathcal{S}$ be the distribution $(\mathbf{x}, f(\mathbf{x}))$ where $\mathbf{x} \xleftarrow{\$} \mathcal{D}$, and let $k = t + 1$. It follows that with probability $1 - 1/10$, it holds that $(\mathbf{x}_{t+1}, f(\mathbf{x}_{t+1})) \in \mathrm{Span}\{\mathbf{v}_i\}_{i\in[t]}$ which implies that $\langle(-\mathbf{s}, 1), (\mathbf{x}_{t+1}, f(\mathbf{x}_{t+1}))\rangle = 0$, or in other words $f(\mathbf{x}_{t+1}) = \langle \mathbf{s}, \mathbf{x}_{t+1}\rangle = h(\mathbf{x}_{t+1})$.

## 3 Homomorphism is a Liability When Decryption is Learnable

This section features our main result. We show that schemes with learnable decryption circuits are very limited in terms of their homomorphic properties, regardless of decryption error. This extends the previous results of [15, 16] showing that the decryption function cannot be learnable if the decryption error is negligible.

We start by showing that a scheme with $(t, 1/10)$-sc-learnable decryption function (i.e. efficient learning with probability $1/10$ using $t$ samples, see Definition 2.6) cannot have decryption error smaller than $\Omega(1/t)$ and be secure (regardless of homomorphism). We proceed to show that if the scheme can homomorphically evaluate the majority function, then the above amplifies dramatically and security cannot be guaranteed for *any* reasonable decryption error ($1/2 - \epsilon$ error for any noticeable $\epsilon$). Using Claim 2 (boosting), this implies that the above hold for any scheme with weakly-learnable (or sc-learnable) decryption. We then discuss the role of key generation error compared to encryption error.

For the sake of simplicity, we focus on the public key setting. However, our proofs easily extend also to symmetric encryption, since our attacks only use the public key in order to generate ciphertexts for known messages.

*Learnable Decryption without Homomorphism.* We start by showing that a scheme whose decryption circuit is $(t, 1/10)$-sc-learnable has to have decryption error $\epsilon = \Omega(1/t)$, otherwise it is insecure. This is a parameterized and slightly generalized version of the claims of [15, 16], geared towards schemes with high decryption error and possibly bad keys. The basic idea is straightforward: We use the public key to generate $t$ ciphertexts to be used as labeled samples for our learner, and then use its output hypothesis to decrypt the challenge ciphertext. The above succeeds so long as all samples in the experiment decrypt correctly, which by the union bound is at least $1 - t \cdot \epsilon$. A formal statement and proof follows.

**Lemma 3.1.** *An encryption scheme whose decryption function is $(t, 1/10)$-sc-learnable for a polynomial $t$ and whose decryption error $< 1/(10(t+1))$ is insecure.*

*Proof.* Consider a key pair $(pk, sk)$ for the scheme, and consider the following CPA adversary. The adversary first generates $t$ labeled samples of the form $(\mathsf{Enc}_{pk}(m), m)$, for random messages $m \xleftarrow{\$} \{0,1\}$ (where $0, 1$ serve as generic names for an arbitrary pair of elements in the scheme's message space). These samples are fed into the aforementioned learner, let $h$ denote the learner's output hypothesis. The adversary lets $m_0 = 0$, $m_1 = 1$, and given the challenge ciphertext $c = \mathsf{Enc}_{pk}(m_b)$, it outputs $b' = h(c)$.

To analyze, we consider the (inefficient) distribution $\mathcal{D}$ that first samples $m \xleftarrow{\$} \{0,1\}$, and then outputs a random *correctly decryptable* encryption of $m$. More formally, $\mathcal{D}$ is the distribution $c = \mathsf{Enc}_{pk}(m) | (\mathsf{Dec}_{sk}(c) = m)$ for a randomly chosen $m \xleftarrow{\$} \{0,1\}$. By Definition 2.6, if the learner gets $t$ samples from this distribution, it outputs a hypothesis that correctly labels the $(t+1)$ sample, with all but $1/10$ probability.

While we cannot efficiently sample from $\mathcal{D}$ (without the secret key), we show that the samples (and challenge) that we feed to our learner are in fact statistically close to samples from $\mathcal{D}$. Consider a case where $(pk, sk)$ are such that the decryption error is indeed smaller than $\epsilon = 1/(10(t+1))$. In such case, our adversary samples from a distribution of statistical distance at most $\epsilon$ from $\mathcal{D}$, and the challenge ciphertext is drawn from the same distribution. It follows that the set of $(t+1)$ samples that we consider during the experiment (containing the labeled samples and the challenge), agree with $\mathcal{D}$ with all but $(t+1) \cdot \epsilon = 1/10$ probability.

Using the union bound on all aforementioned "bad" events (the key pair not conforming with decryption error as per Definition 2.1, the samples not agreeing with $\mathcal{D}$, and the learner failing), we get that $\Pr[b' = b] \geq 1 - 0.01 - 1/10 - 1/10 > 0.7$ and the lemma follows.

Using Claim 2, we derive the following corollary.

**Corollary 3.2.** *An encryption scheme whose decryption function is weakly-learnable must have decryption error $1/\mathrm{poly}(n)$ for some polynomial.*

We note that this corollary does not immediately follow from [15, 16] if a noticeable fraction of the keys can be "bad" (since they do not use boosting).

Plugging our learner for linear functions (Corollary 2.7) into Lemma 3.1 implies the following, which will be useful for the next section.

**Corollary 3.3.** *There exists a constant $\alpha > 0$ such that any $n$-linearly decryptable scheme with decryption error $< \alpha/n$ is insecure.*

*Learnable Decryption with Majority Homomorphism.* Lemma 3.1 and Corollary 3.2 by themselves are not very restrictive. Specifically, they are not directly applicable to attacking any known scheme. Indeed, known schemes with linear

decryption (e.g. LPN based) have sufficiently high decryption error (or, viewed differently, adding the error makes the underlying decryption hard to learn). We now show that if homomorphism is required as a property of the scheme, then decryption error cannot save us.

The following theorem states that majority-homomorphic schemes (see Definition 2.2) cannot have learnable decryption for any reasonable decryption error.

**Theorem 3.4.** *An encryption scheme whose decryption circuit is* $(t, 1/10)$*-sc-learnable for a polynomial* $t$ *and whose decryption error* $< (1/2 - \epsilon)$ *cannot be* $O(\log t/\epsilon^2)$*-majority-homomorphic.*

Let us first outline the proof of Theorem 3.4 before formalizing it. Our goal is the same as in the proof of Lemma 3.1, to generate $t$ labeled samples, which will enable to break security. However, unlike above, taking $t$ random encryptions will surely introduce decryption errors. We thus use the majority homomorphism: We generate a good encryption of $m$, i.e. one that is decryptable with high probability, by generating $O(\log t/\epsilon^2)$ random encryptions of $m$, and apply majority homomorphically. Chernoff's bound guarantees that with high probability, more than half of the ciphertexts are properly decryptable, and therefore the output of the majority evaluation is with high probability a decryptable encryption of $m$. At this point, we can apply the same argument as in the proof of Lemma 3.1. The formal proof follows.

*Proof.* Consider an encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as in the theorem statement. We will construct a new scheme $(\mathsf{Gen}' = \mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}' = \mathsf{Dec})$ (with the same key generation and decryption algorithms) whose security relates to that of $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Then we will use Lemma 3.1 to render the latter scheme insecure.

The new encryption algorithm $\mathsf{Enc}'_{pk}(m)$ works as follows: To encrypt a message $m$, invoke the original encryption $\mathsf{Enc}_{pk}(m)$ for (say) $k = 10(\ln(t+1) + \ln(10))/\epsilon^2$ times, thus generating $k$ ciphertexts. Apply $\mathsf{MajEval}$ to those $k$ ciphertexts and output the resulting ciphertext.

The security of the new scheme is related to that of the original by a straightforward hybrid argument. We will show that the new scheme has decryption error at most $1/(10(t+1))$, but in a slightly weaker sense then Definition 2.1: We will allow 2% of the keys to be "bad" instead of just 1% as before. One can easily verify that the proof of Lemma 3.1 works in this case as well.

Our set of good key pairs for $\mathsf{Enc}'$ is those for which $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\cdot))$ indeed have decryption error at most $1/2 - \epsilon$ and in addition $\mathsf{MajEval}$ is correct. By the union bound this happens with probability at least 0.98.

To bound the decryption error of $\mathsf{Dec}_{sk}(\mathsf{Enc}'_{pk}(\cdot))$, assume that we have a good key pair as described above. We will bound the probability that more than a $1/2 - \epsilon/2$ fraction of the $k$ ciphertexts generated by $\mathsf{Enc}'$ are decrypted incorrectly. Clearly if this bad event does not happen, then by the correctness of $\mathsf{MajEval}$, the resulting ciphertext will decrypt correctly.

Recalling that the expected fraction of falsely decrypted ciphertexts is at most $1/2 - \epsilon$, the Chernoff bound implies that the aforementioned bad event

happens with probability at most

$$e^{-2(\epsilon/2)^2 k} < 1/(10(t+1)) \ ,$$

and the theorem follows.

From the proof it is obvious that even "approximate-majority homomorphism" is sufficient for the theorem to hold. Namely, even if $\mathsf{MajEval}$ only computes the majority function correctly if the fraction of identical inputs is more than $1/2 + \epsilon/2$. Even more generally, we can use any function $\mathsf{FEval}$ for which $\mathsf{Dec}_{sk}(\mathsf{FEval}(\mathsf{Enc}_{pk}(m), \ldots, \mathsf{Enc}_{pk}(m))) = m$ with high probability.

We can derive a general corollary for every weakly-learnable function using Claim 2. This applies, for example, to linear functions, low degree polynomials and shallow circuits.

**Corollary 3.5.** *An encryption scheme whose decryption function is weakly-learnable and whose decryption error is* $1/2 - \epsilon$ *cannot be* $\omega(\log n/\epsilon^2)$*-majority-homomorphic.*

*The Role of Bad Keys.* Recall that in Definitions 2.1 and 2.2 (decryption error and majority homomorphism) we allowed a constant fraction of keys to be useless for the purpose of decryption and homomorphic evaluation, respectively. In fact, it is this relaxation that makes our argument more involved than [15, 16].

As we mentioned above, the choice of constant 0.01 is arbitrary. Let us now explain how our results extend to the case of $1/2 - \kappa$ fraction of bad keys, where $\kappa = 1/\mathrm{poly}(n)$ (we now count the keys that are either bad for decryption or bad for homomorphism). In such case, the argument of Lemma 3.1 will work so long as we start with a $(t, \eta)$-sc-learner with $\eta < \kappa/3$ and so long as the decryption error for good keys is at most $\kappa/(3(t+1))$. If the scheme is furthermore $O(\log(t/\kappa)/\epsilon^2) = O(\log n/\epsilon^2)$-majority-homomorphic, the proof of Theorem 3.4 will also go through. Finally, using boosting, we can start with any weak learner and reduce $\eta$ to $< \kappa/3$ at the cost of a polynomial increase in $t$, which is tolerable by our arguments (and swallowed by the asymptotic notation).

## 4 Attacks on the BL Scheme

In this section we use our tools from above to show that the BL scheme (outlined in Section 4.1 below) is broken. We present two attacks: the first, in Section 4.2, follows from Corollary 3.3 (and works in the spirit of Theorem 3.4); and the second, in Section 4.3, directly attacks a lower level subcomponent of the scheme and allows to decrypt any ciphertext. In fact, the latter attack also follows the same basic principles and exploits a "built-in" evaluation of majority that exists in that sub-component of BL.

### 4.1 Essentials of the BL Scheme

In this section we present the properties of the BL scheme. We concentrate on the properties that are required for our breaks. We refer the reader to [3] for further details.

The BL scheme has a number of layers of abstraction, which are all instantiated based on a global parameter $0 < \alpha < 0.25$ as explained below.

*The Scheme* $\mathbf{K}_q(n)$. BL introduce $\mathbf{K}_q(n)$, a public-key encryption scheme with imperfect correctness. For security parameter $n$, the public key is a matrix $\mathbf{P} \in \mathbb{F}_q^{n \times r}$, where $r = n^{1-\alpha/8}$, and the secret key is a vector $\mathbf{y} \in \mathbb{F}_q^n$ in the kernel of $\mathbf{P}^T$ (namely, $\mathbf{y}^T \cdot \mathbf{P} = 0$). The keys are generated in a highly structured manner in order to support homomorphism, but their structure is irrelevant to us. An encryption of a message $m \in \mathbb{F}_q$ is a vector $\mathbf{c} = \mathbf{P} \cdot \mathbf{x} + m \cdot \mathbf{1} + \mathbf{e}$, where $\mathbf{x} \in \mathbb{F}_q^r$ is some vector, and where $\mathbf{e} \in \mathbb{F}_q^n$ is a low hamming weight vector. Decryption is performed by taking the inner product $\langle \mathbf{y}, \mathbf{c} \rangle$, and succeeds so long as $\langle \mathbf{y}, \mathbf{e} \rangle = 0$ (the vector $\mathbf{y}$ is chosen such that $\langle \mathbf{y}, \mathbf{1} \rangle = 1$). It is shown how the structure of the keys implies that decryption succeeds with probability at least $\left(1 - n^{-(1-\alpha/2)}\right)$. Finally, BL show that $\mathbf{K}_q(n)$ is homomorphic with respect to a single addition or multiplication.[3]

*Re-Encryption.* In order to enable homomorphism, BL introduce the notion of re-encryption. Consider an instantiation of $\mathbf{K}_q(n)$, with keys $(\mathbf{P}, \mathbf{y})$, and an instantiation of $\mathbf{K}_q(n')$ with keys $(\mathbf{P}', \mathbf{y}')$, for $n' = n^{1+\alpha}$. Let $\mathbf{H}_{n':n} \in \mathbb{F}_q^{n' \times n}$ be an element-wise encryption of $\mathbf{y}$ using the public key $\mathbf{P}'$.[4] Namely $\mathbf{H}_{n':n} = \mathbf{P}' \cdot \mathbf{X}' + \mathbf{1} \cdot \mathbf{y}^T + \mathbf{E}'$. Due to the size difference between the schemes, it holds that with probability $\left(1 - n^{-\Omega(1)}\right)$, all of the columns of $\mathbf{H}_{n':n}$ are simultaneously decryptable and indeed $\mathbf{y}'^T \cdot \mathbf{H}_{n':n} = \mathbf{y}^T$. In such case, for any ciphertext $\mathbf{c}$ of $\mathbf{K}_q(n)$, we get $\langle \mathbf{y}', \mathbf{H}_{n':n}\mathbf{c} \rangle = \langle \mathbf{y}, \mathbf{c} \rangle$. The matrix $\mathbf{H}_{n':n}$ therefore re-encrypts ciphertexts of $\mathbf{K}_q(n)$ as ciphertexts of $\mathbf{K}_q(n')$.

The critical idea for our second break is that a re-encrypted ciphertext always belongs to an $n$-dimensional linear subspace (recall that $n \ll n'$), namely to the span of $\mathbf{H}_{n':n}$.

*The Scheme* **BASIC**. Using re-encryption, BL construct a ladder of schemes of increasing lengths that allow for homomorphic evaluation. They define the scheme **BASIC** which has an additional depth parameter $d = O(1)$ (BL suggest to use $d = 8$, but our attack works for any $d > 1$). They consider instantiations of $\mathbf{K}_q(n_i)$, where $n_i = n^{(1+\alpha)^{-(d-i)}}$, for $i = 0, \ldots, d$, so $n_d = n$. They generate

[3] Homomorphic operations (addition, multiplication) are performed element-wise on ciphertext vectors, and the structure of the key guarantees that correctness is preserved.

[4] A note on notation: In [3], the re-encryption parameters are denoted by $I$ (as opposed to our $\mathbf{H}$). We feel that their notation ignores the important linear algebraic structure of the re-encryption parameters, and therefore we switched to matrix notation, which also dictated the change of letter.

all re-encryption matrices $\mathbf{H}_{n_{i+1}:n_i}$ (with success probability $\left(1-n^{-\Omega(1)}\right)$) and can thus homomorphically evaluate depth $d$ circuits.

The homomorphic evaluation works by performing a homomorphic operation at level $i$ of the evaluated circuit (with $i$ going from 0 to $d-1$), and then using re-encryption with $\mathbf{H}_{n_{i+1}:n_i}$ to obtain a fresh ciphertext for the next level.

For the purposes of our (second) break, we notice that in the last step of this evaluation is re-encryption using $\mathbf{H}_{n_d:n_{d-1}}$. This means that homomorphically evaluated ciphertexts all come from a linear subspace of dimension $n_{d-1} = n^{1/(1+\alpha)}$.

*Error Correction and the Matrix* $\mathbf{H}_{n:n}$. Up to this point, BL only get homomorphism at the cost of increasing the input size (namely $n$). In order to get size-preserving homomorphism, BL show that given key-pairs $(\mathbf{P}, \mathbf{y})$, $(\mathbf{P}^*, \mathbf{y}^*)$ for $\mathbf{K}_q(n)$; they can generate, with probability $\left(1-n^{-\Omega(1)}\right)$, a matrix $\mathbf{H}_{n:n}$ whose columns are encryptions of $\mathbf{y}$ under $\mathbf{y}^*$. Most importantly, $\mathbf{H}_{n:n}$ should not give an attacker any excess power. Such a matrix will allow a single size-preserving homomorphic operation.

The idea is to think about $(\mathbf{P}^*, \mathbf{y}^*)$ as the last step of the key ladder in **BASIC**, and generate encryptions of $\mathbf{y}$ under the first step of that ladder. Naturally, the probability that all of those encryptions are simultaneously correctly decryptable is very slim, but the depth $d$ homomorphism of **BASIC** can then be used to homomorphically apply error correction on these ciphertexts. More details follow.

BL generate an instance of **BASIC**, with public keys $\mathbf{P}_0, \ldots, \mathbf{P}_d$, secret key $\mathbf{y}_d = \mathbf{y}^*$, and re-encryption matrices $\mathbf{H}_{n_{i+1}:n_i}$. An additional independent instance of $\mathbf{K}_q(n)$ is generated, whose keys we denote by $(\mathbf{P}, \mathbf{y})$. Then, a large number of encryptions of the elements of $\mathbf{y}$ under public key $\mathbf{P}_0$ are generated.[5] While some of these ciphertexts may have encryption error, BL show that homomorphically evaluating a depth-$d$ correction circuit ($CORR$ in their notation), one can obtain a matrix $\mathbf{H}_{n:n}$, whose columns are encryptions of $\mathbf{y}$ that are decryptable under $\mathbf{y}^*$ without error. This process succeeds with probability $\left(1-n^{-\Omega(1)}\right)$.

The resemblance to the learner of Corollary 2.7 is apparent. In a sense, the public key of **BASIC** is ready-for-use learner.

To conclude this part, BL generate a re-encryption matrix $\mathbf{H}_{n:n}$ that takes ciphertexts under $\mathbf{y}$ and produces ciphertexts under $\mathbf{y}^*$. Since $\mathbf{H}_{n:n}$ is produced using homomorphic evaluation, its rank is at most $n_{d-1} = n^{1/(1+\alpha)}$. We will capitalize on the fact that re-encryption using $\mathbf{H}_{n:n}$ produces ciphertexts that all reside in a low-dimensional space.

*Achieving Full Homomorphism – The Scheme* **HOM**. The basic idea is to generate a sequence of matrices $\mathbf{H}_{n:n}$, thus creating a chaining of the respective secret keys that will allow homomorphism of any depth. However, generating

[5] To be absolutely precise, BL encrypt a bit decomposition of $\mathbf{y}^*$, but this is immaterial to us.

an arbitrarily large number of such re-encryption matrices will eventually cause an error somewhere down the line. Therefore, a more sophisticated solution is required. BL suggest to encrypt each message a large number of times, and generate a large number of re-encryption matrices per level. Then, since the vast majority of matrices per level are guaranteed to be correct, one can use shallow approximate majority computation to guarantee that the fraction of erroneous ciphertexts per level does not increase with homomorphic evaluation.

Decryption is performed as follows: Each ciphertext is a set of ciphertexts $c_1, \ldots, c_k$ of $\mathbf{K}_q(n)$ (all with the same secret key). The decryption process first uses the $\mathbf{K}_q(n)$ key to decrypt the individual ciphertexts and obtain $m_1, \ldots, m_k$, and then outputs the majority between the values $m_i$. BL show that a majority of the ciphertexts (say more than 15/16 fraction) are indeed correct, which guarantees correct decryption.

BL can thus achieve a (leveled) fully homomorphic scheme which they denote by **HOM**, which completes their construction.

### 4.2 An Attack on BL Using Homomorphism

We will show how to break the BL scheme using its homomorphic properties. We use Corollary 3.3 and our proof contains similar elements to the proof of Theorem 3.4. (The specifics of BL do not allow to use Corollary 3.5 directly.)

**Theorem 4.1.** *There is a polynomial time CPA attack on BL.*

*Proof.* Clearly we cannot apply our methods to the scheme **HOM** as is, since its decryption is not learnable. We thus describe a related scheme which is "embedded" in **HOM** and show how to distinguish encryptions of 0 from encryptions of 1, which will imply a break of **HOM**.

We recall that the public key of **HOM** contains "chains" of re-encryption matrices of the form $\mathbf{H}_{n:n}$. The length of the chains is related to the homomorphic depth of **HOM**. Our sub-scheme will only require a chain of constant length $\ell$ which will be determined later (such sub-chain therefore must exist for any instantiation of BL that allows for more than constant depth homomorphism). Granted that all links in the chain are successfully generated (which happens with probability $\ell \cdot n^{-\Omega(1)}$), such a chain allows homomorphic evaluation of any depth-$\ell$ function. Let us focus on the case where the chain is indeed properly generated.

Intuitively, we would have liked to use this structure to evaluate majority on $2^\ell$ input ciphertexts. However, BL is defined over a large field $\mathbb{F}$, and it is not clear how to implement majority over $\mathbb{F}$ in depth that does not depend on $q = |\mathbb{F}|$. To solve this problem, we use BL's $CORR$ function. This function is just a NAND tree of depth $\ell$ (extended to $\mathbb{F}$ in the obvious way: $NAND(x, y) = 1 - xy$). BL show that given $2^\ell$ inputs, each of which is 0 (respectively 1) with probability $1 - \epsilon$, the output of $CORR$ will be 0 (resp. 1) with probability $1 - O(\epsilon)^{2^{\ell/2}}$.

To encrypt a message $m \in \{0, 1\}$ using our sub-scheme, we will generate $2^\ell$ ciphertexts. Each ciphertext will be an independent encryption of $m$ using the

public key of **HOM** (which essentially generates $\mathbf{K}_q(n)$ ciphertexts that correspond to the first link in all chains). We then apply $CORR$ homomorphically to the generated ciphertexts. Decryption in our subscheme will be standard $\mathbf{K}_q(n)$ decryption (which is a linear function) using the secret key that corresponds to the last link in the chain.[6]

We recall that the decryption error of $\mathbf{K}_q(n)$ is $\epsilon = n^{-\Omega(1)}$. By the properties of $CORR$, we can choose $\ell = O(1)$ such that the decryption error of our subscheme is at most (say) $o(1)/n$.

In conclusion, we get a sub-scheme of **HOM** such that with probability $1 - n^{-\Omega(1)} > 0.9$ over the key generation, the decryption error is at most $o(1)/n$. Furthermore, decryption is linear. Corollary 3.3 implies that such scheme must be insecure.

### 4.3 A Specific Attack on BL

We noticed that the scheme **BASIC**, which is a component of **HOM**, contains by design homomorphic evaluation of majority: this is how the matrix $\mathbf{H}_{n:n}$ is generated. We thus present an attack that only uses the matrix $\mathbf{H}_{n:n}$ and allows to completely decrypt BL ciphertexts (even non binary) with probability $1 - n^{-\Omega(1)}$. We recall that an attack *completely breaks* a scheme if it can decrypt any given ciphertext with probability $1 - o(1)$.

**Theorem 4.2.** *There exists a polynomial time attack that completely breaks* **BASIC***, and thus also BL.*

*Proof.* We consider the re-encryption matrix $\mathbf{H} = \mathbf{H}_{n:n} \in \mathbb{F}_q^{n \times n}$ described in Section 4.1, which re-encrypts ciphertexts under $\mathbf{y}$ into ciphertexts under $\mathbf{y}^*$. The probability that $\mathbf{H}$ was successfully generated is at least $1 - n^{-\Omega(1)}$, in which case it holds that

$$\mathbf{y}^{*T} \cdot \mathbf{H} = \mathbf{y}^T \ .$$

In addition, as we explained in Section 4.1, the rank of $\mathbf{H}$ is at most $h = n^{1/(1+\alpha)}$.

Our breaker will be given $\mathbf{H}$ and the public key $\mathbf{P}$ that corresponds to $\mathbf{y}$, and will be able to decrypt any vector $\mathbf{c} = \mathsf{Enc}_{\mathbf{P}}(m)$ with high probability, namely compute $\langle \mathbf{y}, \mathbf{c} \rangle$.

*Breaker Code.* As explained above, the input to the breaker is $\mathbf{H}, \mathbf{P}$ and challenge $\mathbf{c} = \mathsf{Enc}_{\mathbf{P}}(m)$. The breaker will execute as follows:

1. Generate $k = h^{1+\epsilon}$ encryptions of 0, denoted $\mathbf{v}_1, \ldots, \mathbf{v}_k$, for $\epsilon = \frac{\alpha(1-\alpha)}{4}$ (any positive number smaller than $\frac{\alpha(1-\alpha)}{2}$ will do).
   Note that this means that with probability $1 - n^{-\Omega(1)}$, all $\mathbf{v}_i$ are decryptable encryptions of 0. Intuitively, these vectors, once projected through $\mathbf{H}$, will span all decryptable encryptions of 0.

[6] The secret key of the last link is not the same as the secret key of **HOM**, since we are only considering a sub-chain of a much longer chain. However, this is not a problem: Our arguments do not require that the secret key is known to anyone in order to break the scheme.

2. For all $i = 1, \ldots, k$, compute $\mathbf{v}_i^* = \mathbf{H} \cdot \mathbf{v}_i$ (the projections of the ciphertexts above through $\mathbf{H}$). Also compute $\mathbf{o}^* = \mathbf{H} \cdot \mathbf{1}$ (the projection of the all-one vector).
3. Find a vector $\tilde{\mathbf{y}}^* \in \mathbb{F}_q^n$ such that $\langle \tilde{\mathbf{y}}^*, \mathbf{v}_i^* \rangle = 0$ for all $i$, and such that $\langle \tilde{\mathbf{y}}^*, \mathbf{o}^* \rangle = 1$. Such a vector necessarily exists if all $\mathbf{v}_i$'s are decryptable, since $\mathbf{y}^*$ is an example of such a vector.
4. Given a challenge ciphertext $\mathbf{c}$, compute $\mathbf{c}^* = \mathbf{H} \cdot \mathbf{c}$ and output $m = \langle \tilde{\mathbf{y}}^*, \mathbf{c}^* \rangle$ (namely, $m = \tilde{\mathbf{y}}^{*T} \cdot \mathbf{H} \cdot \mathbf{c}$).

*Correctness.* To analyze the correctness of the breaker, we first notice that the space of ciphertexts that decrypt to 0 under $\mathbf{y}$ is linear (this is exactly the orthogonal space to $\mathbf{y}$). We denote this space by $Z$. Since $\mathbf{1} \notin Z$, we can define the cosets $Z_m = Z + m \cdot \mathbf{1}$. We note that all legal encryptions of $m$ using $\mathbf{P}$ reside in $Z_m$.

We let $Z^*$ denote the space $\mathbf{H} \cdot Z$ (all vectors of the form $\mathbf{H} \cdot \mathbf{z}$ such that $\mathbf{z} \in Z$). This is a linear space with dimension at most $h$. Similarly, define $Z_m^* = Z^* + m \cdot \mathbf{o}^*$.

Consider the challenge ciphertext $\mathbf{c} = \mathsf{Enc}_{\mathbf{P}}(m)$. We can think of $\mathbf{c}$ as an encryption of 0 with an added term $m \cdot \mathbf{1}$. We therefore denote $\mathbf{c} = \mathbf{c}_0 + m \cdot \mathbf{1}$. Again this yields a $\mathbf{c}_0^*$ such that $\mathbf{c}^* = \mathbf{c}_0^* + m \cdot \mathbf{o}^*$.

Now consider the distribution $\mathcal{Z}$ over $Z$, which is the distribution of decryptable encryptions of 0 (i.e. the distribution $\mathbf{c} = \mathsf{Enc}_{\mathbf{P}}(0)$, conditioned on $\langle \mathbf{y}, \mathbf{c} \rangle = 0$). The distribution $\mathcal{Z}^*$ is defined by projecting $\mathcal{Z}$ through $\mathbf{H}$. With probability $\left(1 - n^{-\Omega(1)}\right)$, it holds that $\mathbf{v}_1^*, \ldots, \mathbf{v}_k^*$, and $\mathbf{c}_0^*$ are uniform samples from $\mathcal{Z}^*$.

By Lemma 2.4 below, it holds that $\mathbf{c}_0^* \in \mathrm{Span}\{\mathbf{v}_1^*, \ldots, \mathbf{v}_k^*\}$, with probability $\left(1 - n^{-\Omega(1)}\right)$. In such case

$$\langle \tilde{\mathbf{y}}^*, \mathbf{c}^* \rangle = \langle \tilde{\mathbf{y}}^*, \mathbf{c}_0^* \rangle + m \cdot \langle \tilde{\mathbf{y}}^*, \mathbf{o}^* \rangle = m \ .$$

We conclude that with probability $1 - n^{-\Omega(1)}$, our breaker correctly decrypts $\mathbf{c}$ as required.

**Acknowledgements**

The author wishes to thank the MIT-BU cryptography reading group for introducing the BL scheme to him, and especially to Stefano Tessaro and Shafi Goldwasser for various discussions. We further thank Andrej Bogdanov for his comments on a preprint of this manuscript, and for the discussions that followed, leading to the general argument about homomorphic schemes. We thank Ron Rothblum for pointing out the learning-theory aspect of our argument. Lastly, we thank the anonymous reviewers of TCC 2013 for their helpful comments.

## References


[1] M. Alekhnovich. More on average case vs approximation complexity. In *FOCS*, pages 298–307. IEEE Computer Society, 2003.

[2] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009.

[3] A. Bogdanov and C. H. Lee. Homomorphic encryption from codes. Cryptology ePrint Archive, Report 2011/622, 2011. `http://eprint.iacr.org/`.

[4] Z. Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012. See also `http://eprint.iacr.org/2012/078`.

[5] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. ITCS, 2012. See also `http://eprint.iacr.org/2011/277`.

[6] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, volume 6841, page 501, 2011.

[7] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Ostrovsky [17], pages 97–106. References are to full version: `http://eprint.iacr.org/2011/344`.

[8] V. Gauthier, A. Otmani, and J.-P. Tillich. A distinguisher-based attack of a homomorphic encryption scheme relying on reed-solomon codes. Cryptology ePrint Archive, Report 2012/168, 2012. `http://eprint.iacr.org/`.

[9] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[10] C. Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *CRYPTO*, pages 116–137, 2010.

[11] C. Gentry and S. Halevi. Implementing gentry's fully-homomorphic encryption scheme. In K. G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148. Springer, 2011.

[12] C. Gentry, S. Halevi, and N. P. Smart. Fully homomorphic encryption with polylog overhead. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2012.

[13] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. How to encrypt with the lpn problem. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 679–690. Springer, 2008.

[14] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In H. R. Lewis, B. B. Simons, W. A. Burkhard, and L. H. Landweber, editors, *STOC*, pages 365–377. ACM, 1982.

[15] M. J. Kearns and L. G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994. Preliminary version in STOC 89.

[16] A. R. Klivans and A. A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. *J. Comput. Syst. Sci.*, 75(1):2–12, 2009. Preliminary version in FOCS 06.

[17] R. Ostrovsky, editor. *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*. IEEE, 2011.

[18] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990. Preliminary version in FOCS 89.

[19] N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In P. Q. Nguyen and D. Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer, 2010.

[20] V. Vaikuntanathan. Computing blindfolded: New developments in fully homomorphic encryption. In Ostrovsky [17], pages 5–16.

[21] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. Preliminary version in STOC 84.

[22] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010. Full Version in `http://eprint.iacr.org/2009/616.pdf`.