Incremental Deterministic Public-Key Encryption

Ilya Mironov* Omkant Pandey† Omer Reingold* Gil Segev*

Abstract

Motivated by applications in large storage systems, we initiate the study of incremental deterministic public-key encryption. Deterministic public-key encryption, introduced by Bellare, Boldyreva, and O'Neill (CRYPTO '07), provides a realistic alternative to randomized public-key encryption in various scenarios where the latter exhibits inherent drawbacks. A deterministic encryption algorithm, however, cannot satisfy any meaningful notion of security for low-entropy plaintexts distributions, and Bellare et al. demonstrated that a strong notion of security can in fact be realized for relatively high-entropy plaintext distributions.

In order to achieve a meaningful level of security, a deterministic encryption algorithm should be typically used for encrypting rather long plaintexts for ensuring a sufficient amount of entropy. This requirement may be at odds with efficiency constraints, such as communication complexity and computation complexity in the presence of small updates. Thus, a highly desirable property of deterministic encryption algorithms is incrementality: small changes in the plaintext translate into small changes in the corresponding ciphertext.

We present a framework for modeling the incrementality of deterministic public-key encryption. Within our framework we propose two schemes, which we prove to enjoy an optimal tradeoff between their security and incrementality up to small polylogarithmic factors. Our first scheme is a generic method which can be based on any deterministic public-key encryption scheme, and in particular, can be instantiated with any semantically-secure (randomized) public-key encryption scheme in the random oracle model. Our second scheme is based on the Decisional Diffie-Hellman assumption in the standard model.

The approach underpinning our schemes is inspired by the fundamental "sample-then-extract" technique due to Nisan and Zuckerman (JCSS '96) and refined by Vadhan (J. Cryptology '04), and by the closely related notion of "locally-computable extractors" due to Vadhan. Most notably, whereas Vadhan used such extractors to construct *private-key* encryption schemes in the bounded-storage model, we show that techniques along these lines can also be used to construct incremental *public-key* encryption schemes.

A preliminary version of this work appears in *Advances in Cryptology – EUROCRYPT '12*, pages ???–???, 2012.

*Microsoft Research Silicon Valley, Mountain View, CA 94043, USA. Email: `{mironov,omer.reingold,gil.segev}@microsoft.com`.

†Microsoft, Redmond (USA) and Microsoft Research, Bangalore (India). Email: `omkantp@microsoft.com`.

1 Introduction

The fundamental notion of *semantic security* for public-key encryption schemes was introduced by Goldwasser and Micali [GM84]. While semantic security provides strong privacy guarantees, it inherently requires a *randomized* encryption algorithm. Unfortunately, randomized encryption breaks several assumptions of large storage systems that are crucial in efficient implementation of search (and, more generally, of indexing) and de-duplication [BCG+00, QD02]. Further, randomized encryption necessarily expands the length of the plaintext, which may be undesirable in some applications, such as legacy code or in-place encryption.

Deterministic encryption. To deal with these and other drawbacks, Bellare, Boldyreva, and O'Neill [BBO07] initiated the study of deterministic public-key encryption schemes. These are public-key encryption schemes where the encryption algorithm is deterministic. Bellare et al. formulate meaningful, and essentially "best possible", security requirements for such schemes which are inspired by and very close to semantic security. Clearly, in this setting, no meaningful notion of security can be achieved if the space of plaintexts is small. Therefore, Bellare et al. [BBO07] required security to hold only when the plaintexts are drawn from a high min-entropy distribution.

Deterministic encryption already alleviates many of the above mentioned problems when dealing with large data volumes. For example, since the encryption algorithm is deterministic, we can now do indexing and perform fast search on encrypted data. Further, schemes that have length-preserving ciphertexts are possible as well [BBO07]. Also, unlike randomized encryption, there is no fundamental reason that precludes noticeable savings in storage by using de-duplication techniques (which can be as large as 97% [ZLP08]); although one may not get the same amount of savings as with usual plaintext.

We emphasize that security of deterministic encryption is contingent on a very strong assumption about the underlying data distribution, namely that the plaintext has high min-entropy from the adversary's point of view. One possibility for improving security margin is to encrypt longer plaintexts whenever possible, for example, by not cutting files into smaller pieces or using larger blocks for in-place encryption. If, however, changing the plaintext requires re-computation of the ciphertext, doing that for any update may quickly negate all efficiency gains from using deterministic encryption. For a remedy we turn to *incremental cryptography*, explained below.

Incremental cryptography. Given that we are dealing with large plaintexts, computing the ciphertext from scratch for the modified plaintext can be quite an expensive operation. One such example is maintaining an (encrypted) daily back-up of your hard-disk on an untrusted server. The disk may contain gigabytes of data, most of which is likely to remain unchanged between two successive back-ups. The problem is further intensified in various client-server settings where *all* of previous plaintext might not be available when the modification request is made. In such settings where plaintext is really large, downloading old data can be a serious problem. This issue is clearly not specific to (deterministic) encryption, and is of very general interest.

To address this issue, Bellare, Goldreich and Goldwasser [BGG94] introduced and developed the notion of *incremental* cryptography, first in application to digital signatures. The idea is that, once we have signed a document M, signing new versions of M should be rather quick. For example, if we only flip a single bit of M, we should be able to update the signature in time polynomial in $\log |M|$ (instead of $|M|$) and the security parameter λ. Clearly, incrementality is an attractive feature to have for any cryptographic primitive such as encryption, signatures, hash functions, and so on [BGG95, Mic97, Fis97a, BM97, BKY01].

It is clear from our discussion that when dealing with deterministic encryption over large databases, where we are forced to encrypt rather long plaintexts for ensuring their min-entropy, what we really need is an *incremental* encryption scheme. That is, the scheme should allow quickly updating the ciphertexts to reflect small changes. In light of the observation that deterministic encryption is most desirable when dealing with large data volumes, perhaps it is not exaggerating to suggest that incrementality should be an important *design goal* for deterministic encryption rather than merely a "nice to have" feature.

1.1 Our Contributions

In this work we formalize the notion of *incremental* deterministic public-key encryption. We view incrementality and security as two orthogonal objectives, which together have a great potential in improving the deployment of deterministic encryption schemes with provable security properties in real-world applications.

Modeling incremental updates. Intuitively, a deterministic public-key encryption scheme is *incremental* if any small modification of a plaintext m resulting in a plaintext m' can be efficiently carried over for updating the encryption $c = \mathsf{Enc}_{pk}(m)$ of m to the encryption $c' = \mathsf{Enc}_{pk}(m')$ of m'. For capturing the efficiency of such an update operation we consider two natural complexity measures: (1) *input locality* (i.e., the number of ciphertexts bits that are affected when flipping a single plaintext bit), and (2) *query complexity* (i.e., the number of public-key, plaintext, and ciphertext bits that have to be read in order to update the ciphertext).

We note that modeling updates for deterministic public-key encryption is slightly different than for other primitives. For example, suppose that we allow "replacements" as considered by [BGG94]. These are queries of the form (j, b) that replace the j-th bit of a given plaintext m by $b \in \{0, 1\}$. Then, if there exists a public algorithm Update for updating the ciphertext, then one can recover the entire plaintext from the ciphertext[1]. Therefore, we focus on the *bit flipping* operation instead. This operation is specified by an index j, and sets the current value of $m[j]$ to $\neg m[j]$.

For capturing the above measures of efficiency we model the update operation as a probabilistic polynomial-time algorithm Update that receives as input the index i^* of a plaintext bit to be flipped, and has oracle access to the individual bits of the public key pk, the plaintext m to be modified, and to its encryption $c = \mathsf{Enc}_{pk}(m)$. That is, the algorithm Update can submit queries of the form (pk, i), (m, i) or (c, i), which are answered with the ith bit of pk, m, or c, respectively. We refer the reader to Section 3 for the formal description of our model, which considers also update in a "private" fashion in which the update algorithm can access the secret key but not the plaintext.

Locality lower bound. An important insight is that deterministic encryption cannot have very small incrementality. Deterministic encryption schemes require high min-entropy messages to provide any meaningful guarantee, and we show that any scheme with low incrementality can be secure only for messages with much higher entropy. Specifically, we show that for every deterministic public-key encryption scheme that satisfies the minimal notion of PRIV1-IND security for plaintext distributions of min-entropy k, plaintext length n, and ciphertext length t, the incrementality Δ of the scheme must satisfy:

$$\Delta \geq \frac{n-3}{k \log t} \ .$$

Ignoring the lower-order $\log t$ factor, our proof shows in particular that the input locality of the encryption algorithm must be roughly n/k. This should be compared with the case of randomized encryption, where flipping a single plaintext bit may require to flip only a single ciphertext bit.

[1] The encryption algorithm is deterministic, and hence the ciphertext for every message is unique. The operation $\mathsf{Update}(j, 0)$ changes the ciphertext if and only if the jth bit of m is 1.

Indeed, consider encrypting a plaintext m as the pair $(\mathsf{Enc}_{pk}(r), r \oplus m)$ for a randomly chosen mask r. Flipping a single bit of m requires flipping only a single bit of the ciphertext.

Constructions with optimal incrementality. We construct two deterministic public-key encryption schemes with optimal incrementality (up to lower-order polylogarithmic factors). Our first construction is a general transformation from any deterministic encryption scheme to an incremental one. Following the terminology developed in [BBO07, BFO$^+$08a, BFO08b], the resulting scheme from this approach is PRIV1-IND secure if the underlying scheme is PRIV-IND secure. As a result, using the construction of Bellare et al. [BBO07] in the random oracle model, we can instantiate our approach in the random oracle model based on any semantically-secure (randomized) public-key encryption scheme, and obtain a deterministic scheme with optimal incrementality.

Our second, more direct construction, avoids the random oracle model. It is based on the Decisional Diffie-Hellman assumption in the standard model, and enjoys optimal incrementality. The scheme relies on the notion of *smooth trapdoor functions* that we introduce (and was implicitly used by Boldyreva et al. [BFO08b]), and realize it in an incremental manner based on the Decisional Diffie-Hellman assumption. Both of our constructions guarantee PRIV1-IND security when encrypting n-bit plaintexts with min-entropy $k \geq n^{\epsilon}$, where $\epsilon > 0$ is any pre-specified constant.

1.2 Related Work

The problem of composing public-key encryption and de-duplication was addressed by Doucer et al. [DAB$^+$02] via the concept of *convergent encryption*, in which files are encrypted using their own hash values as keys. Security of the scheme is argued in the random-oracle model and under implicit assumption of the plaintext's high min-entropy. The formal goal of leveraging entropy of the source to achieve information-theoretic security with a short symmetric key was articulated by Russell and Wang [RW02], followed by Dodis and Smith [DS05].

The notion of public-key deterministic encryption was introduced by Bellare, Boldyreva, and O'Neill [BBO07], and then further studied by Bellare, Fischlin, O'Neill, and Ristenpart [BFO$^+$08a], Boldyreva, Fehr, and O'Neill [BFO08b], Brakerski and Segev [BS11], Wee [Wee12], and Fuller, O'Neill and Reyzin [FOR12]. Bellare et al. [BBO07] proved their constructions in the random oracle model; subsequent papers demonstrated schemes secure in the standard model based on trapdoor permutations [BFO$^+$08a] and lossy trapdoor functions [BFO08b]. Brakerski and Segev [BS11] and Wee [Wee12] address the question of security of public-key deterministic encryption in the presence of auxiliary input. Fuller et al. [FOR12] presented a construction based on any trapdoor function that admits a large number of simultaneous hardcore bits, and a construction that is secure for a bounded number of possibly related plaintexts.

Constructions of deterministic public-key encryption found an intriguing application in "hedged" public-key encryptions [BBN$^+$09]. These schemes remain secure even if the randomness used during the encryption process is not perfect (controlled by or leaked to the adversary) as long as the joint distribution of plaintext-randomness has sufficient min-entropy.

The concept of incremental cryptography started with the work of Bellare, Goldreich, and Goldwasser [BGG94], who considered the case of hashing and signing. They also provided discrete-logarithm based constructions for incremental collision-resistant hash and signatures, that support block *replacement* operation. Constructions supporting block *insertion* and *deletion* were first developed in [BGG95], with further refinements and new issues concerning incrementality such as tamper-proof updates, privacy of updates, and incrementality in symmetric encryption. In subsequent work, Fischlin presented an incremental signature schemes supporting insertion/deletion of blocks, and tamper-proof updates [Fis97a], and proved a $\Omega(\sqrt{n})$ lower bound on the signature size of schemes that support substitution and replacement operations (the bound can be improved to $\Omega(n)$

in certain special cases) [Fis97b]. Bellare and Micciancio [BM97] revisited the case of hashing, and provided new constructions for the same based on discrete logarithms and lattices. Buonanno, Katz, and Yung [BKY01] considered the issue of incrementality in symmetric unforgeable encryption and suggested three modes of operations for AES achieving this notion.

The goal of incremental cryptography, i.e., *input locality*, can be contrasted with the dual question of placing cryptography in the NC^0 complexity class, i.e., identifying cryptographic primitives with constant *output locality*. This problem has essentially been resolved for public-key encryption in the positive by Applebaum, Ishai, and Kushilevitz [AIK06], who construct schemes based on standard number-theoretic assumptions and lattice problems where each bit of the encryption operation depends on at most four bits of the input. Applebaum et al. also argue impossibility of semantically-secure public-key encryption scheme with constant input locality [AIK06, Section C.1].

1.3 Overview of Our Approach

In this section we present a high-level overview of our two constructions. First, we describe the well-known "sample-then-extract" approach [NZ96, Vad04] that serves as our inspiration for constructing incremental schemes. Then, we describe the main ideas underlying our schemes, each of which is based on a different realization of the "sample-then-extract" approach.

"Sample-then-extract". A fundamental fact in the theory of pseudorandomness is that a random sample of bits from a string of high min-entropy essentially preserves the min-entropy rate. This was initially proved by Nisan and Zuckerman [NZ96] and then refined by Vadhan [Vad04] that captured the optimal parameters. Intuitively, the "sample-then-extract" lemma states that if $\mathcal{X} \in \{0,1\}^n$ has min-entropy rate δ, and $\mathcal{X}_S \in \{0,1\}^t$ is the projection of $\mathcal{X}$ onto a random set $S \subseteq [n]$ of t positions, then $\mathcal{X}_S$ is statistically-close to a source with min-entropy rate $\delta' = \Omega(\delta)$.

This lemma serves as a fundamental tool in the design of randomness extractors. Moreover, in the cryptographic setting, it was used by Vadhan [Vad04] to construct *locally-computable extractors*, which allow to compute their output by examining a small number of input bits. Such extractors were used by Vadhan to design private-key encryption schemes in the bounded-storage model. In this work we demonstrate for the first time that the "sample-then-extract" approach can be leveraged to design not only *private-key* encryption schemes, but also *public-key* encryption schemes.

A generic construction via random partitioning. In the setting of randomized encryption, a promising approach for ensuring incrementality is to divide each plaintext m into consecutive and rather small blocks $m = m_1||\cdots||m_\ell$, and to separately encrypt each block m_i. Thus, changing a single bit of m affects only a single block of the ciphertext. Moreover, the notion of semantic security is sufficiently powerful to even allow each block m_i to be as small as a single bit. In the setting of deterministic encryption, however, security can hold only when each encrypted block has a sufficient amount of min-entropy. At this point we note that even if a plaintext $m = m_1||\cdots||m_\ell$ has high min-entropy, it may clearly be the case that some of its small blocks have very low min-entropy (or even fixed). Thus, this approach seems to fail for deterministic encryption.

As an alternative, however, we propose the following approach: instead of dividing the plaintext m into fixed blocks, we project it onto a *uniformly chosen partition* $S_1, \ldots, S_\ell$ of the plaintext positions to sets of equal sizes, and then separately encrypt each of the projections $m_{S_1}, \ldots, m_{S_\ell}$ using an underlying (possibly non-incremental) deterministic encryption scheme[2]. By the fact that we use a partition of the plaintext positions we ensure on the one hand that the plaintext m can be fully recovered, and on the other that each plaintext position appears in only one set (and thus

[2] A minor technical detail is that we would also like to ensure that we always encrypt distinct values, and therefore we concatenate the block number i to each projection m_{S_i}.

the scheme is incremental). In terms of security, since we use a uniformly chosen partition, the distribution of each individual set S_i is uniform, and therefore by carefully choosing the size of the sets the "sample-and-extract" lemma guarantees that with overwhelming probability each projection m_{S_i} preserves the min-entropy rate of m. Therefore, the scheme is secure as long as the underlying scheme guarantees PRIV-IND security (see Section 2.2 for the notions of security for deterministic encryption).

By instantiating this approach with the constructions of Bellare et al. [BBO07] in the random oracle model, we obtain as a corollary a deterministic public-key encryption scheme with optimal incrementality based either on any semantically-secure (randomized) public-key encryption scheme, or on RSA-OAEP which yields a *length-preserving* incremental scheme.

A construction based on smooth trapdoor functions. Although our first construction is a rather generic one, constructions of PRIV-IND-secure schemes are known only in the random oracle model. In the standard model, Boldyreva et al. [BFO08b] introduced the slightly weaker notion of PRIV1-IND security, which considers plaintexts that have high min-entropy even when conditioned on other plaintexts, and showed that it can be realized by composing any lossy trapdoor function with a pairwise independent permutation. This approach, however, does not seem useful for constructing incremental schemes, since pairwise independence is inherently non-incremental. A simple observation, however, shows that the approach of Boldyreva et al. [BFO08b] requires in fact trapdoor functions with weaker properties, that we refer to as *smooth trapdoor functions* (this is implicit in [BFO08b]).

Informally, a collection of smooth trapdoor functions consists of two families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor. Functions in the other family are "smooth" in the sense that their output distribution on any source of input with high min-entropy is statistically close to their output distribution on a uniformly sampled input. The only security requirement is that a description of a randomly chosen function from the family of injective functions is computationally indistinguishable from a description of a randomly chosen function from the family of smooth functions. We show that any collection of smooth trapdoor functions is a PRIV1-IND-secure deterministic encryption scheme (again, this is implicit in [BFO08b]).

Next, we construct a collection of *incremental* smooth trapdoor functions based on the Decisional Diffie-Hellman (DDH) assumption, by significantly refining the DDH-based lossy trapdoor functions of Freeman et al. [FGK$^+$10] (which in turned generalized those of Peikert and Waters [PW08]). Our collection is parameterized by a group G of prime order p that is generated by an element $g \in G$. A public key is of the form g^A, where $A \in \mathbb{Z}^{n\times n}$ is sampled from one distribution for injective keys, and from a different distribution for smooth keys[3]. Evaluating a function on an input $x \in \{0,1\}^n$ is done by computing $g^{Ax} \in G^n$ and inversion for injective keys is done using the secret key A^{-1}.

The key point in our scheme is the distribution of the matrix A for injective and smooth keys. For smooth keys the matrix A is generated to satisfy two properties. The first is that each of its first ℓ rows has t randomly chosen entries with values that are chosen uniformly from $\mathbb{Z}_p$, and all other $n-t$ entries are zeros (where ℓ and t are carefully chosen depending on the min-entropy rate). Looking ahead, when computing the inner product of such a sparse row with a source of min-entropy larger than $\log p$, the "sample-then-extract" lemma guarantees that the output is statistically close to uniform. In a sense, this is a realization of a locally-computable extractor that is embedded in our functions. The second property, is that each of its last $n-\ell$ rows are linear combinations of the first ℓ rows, and therefore the image of its corresponding linear map is determined by the first ℓ rows. This way, we can argue that smooth keys hide essentially all information on the underlying input distribution.

[3]For any matrix $A = \{a_{ij}\}_{i\in[n],j\in[n]} \in \mathbb{Z}_p^{n\times n}$ we denote by $g^A \in G^{n\times n}$ the matrix $\{g^{a_{ij}}\}_{i\in[n],j\in[n]}$.

For injective keys, we sample a matrix A from the distribution of smooth keys, and then re-sample all its non-zero entries with independently and uniformly distributed elements of $\mathbb{Z}_p$. A subtle complication arises since such a matrix is not necessarily invertible, as required for injective keys, but this is easily resolved (without hurting the smooth keys – see Section 5 for more details). Observing that for injective keys each column of A contains roughly t non-zero entries, this yields a PRIV1-IND-secure scheme with optimal incrementality.

1.4 Paper Organization

The remainder of this paper is organized as follows. In Section 2 we introduce the notation, tools, and computational assumptions that are used in this paper. In Section 3 we present a framework for modeling the incrementality of deterministic public-key encryption schemes. In Section 4 we present our generic construction, and in Section 5 we present we present our DDH-based construction. Finally, in Section 6 we present the lower bound.

2 Preliminaries

In this section we present the basic notions, definitions, and tools that are used in this paper.

2.1 Probability Distributions

For a distribution $\mathcal{X}$ we denote by $x \leftarrow \mathcal{X}$ the process of sampling a value x according to $\mathcal{X}$. Similarly, for a set Ω we denote by $\omega \leftarrow \Omega$ the process of sampling a value ω from the uniform distribution over Ω. If $\mathcal{X}$ is a distribution and f is a function defined over its support, then $f(\mathcal{X})$ denotes the outcome of the experiment where $f(x)$ is evaluated on x sampled from $\mathcal{X}$. For any $n \in \mathbb{N}$ we denote by $\mathcal{U}_n$ the uniform distribution over the set $\{0,1\}^n$.

The *min-entropy* of a distribution $\mathcal{X}$ that is defined over a set Ω is defined as $\mathrm{H}_\infty(\mathcal{X}) = \min_{\omega\in\Omega} \log\left(1/\Pr[\mathcal{X}=\omega]\right)$. A *k-source* is distribution $\mathcal{X}$ with $\mathrm{H}_\infty(\mathcal{X}) \geq k$, and the *min-entropy rate* of a k-source over the set $\{0,1\}^n$ is k/n. The *statistical distance* between two distributions $\mathcal{X}$ and $\mathcal{Y}$ over a set Ω is defined as $\mathrm{SD}(\mathcal{X},\mathcal{Y}) = \max_{S\subseteq\Omega} |\Pr[\mathcal{X}\in S] - \Pr[\mathcal{Y}\in S]|$. A distribution $\mathcal{X}$ is *ϵ-close* to a k-source if there exists a k-source $\mathcal{Y}$ such that $\mathrm{SD}(\mathcal{X},\mathcal{Y}) \leq \epsilon$. The following standard lemma (see, for example, [DOR+08]) essentially states that revealing r bits of information on a random variable may reduce its min-entropy by roughly r.

Lemma 2.1. *Let $\mathcal{Z}$ be a distribution over at most 2^r values, then for any distribution $\mathcal{X}$ and for any $\epsilon > 0$ it holds that*

$$\Pr_{z\leftarrow\mathcal{Z}}[\mathrm{H}_\infty(\mathcal{X}|\mathcal{Z}=z) \geq \mathrm{H}_\infty(\mathcal{X}) - r - \log(1/\epsilon)] \geq 1-\epsilon \ .$$

We say that two families of distributions $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ are *statistically close*, denoted by $\mathcal{X} \approx \mathcal{Y}$, if there exists a negligible function $\nu(\lambda)$ such that $\mathrm{SD}(\mathcal{X},\mathcal{Y}) \leq \nu(\lambda)$ for all sufficiently large $\lambda \in \mathbb{N}$. Two families of distributions $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ are *computationally indistinguishable*, denoted by $\mathcal{X} \stackrel{c}{\approx} \mathcal{Y}$, if for any probabilistic polynomial-time algorithm A there exists a negligible function $\nu(\lambda)$ such that

$$\left|\Pr_{x\leftarrow\mathcal{X}_\lambda}\left[A(1^\lambda,x)=1\right] - \Pr_{y\leftarrow\mathcal{Y}_\lambda}\left[A(1^\lambda,y)=1\right]\right| \leq \nu(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$.

The "sample-then-extract" lemma. The following lemma due to Vadhan [Vad04] plays a major role in our constructions. This is a refinement of the fundamental "sample-then-extract" lemma that was originally proved by Nisan and Zuckerman [NZ96], stating that a random of sample of bits from a string essentially preserves its min-entropy rate. Vadhan's refinement shows that the min-entropy rate is in fact preserved up to an arbitrarily small additive loss, whereas the original lemma loses a logarithmic factor. Intuitively, the lemma states that if $\mathcal{X} \in \{0,1\}^n$ is a δn-source, and $\mathcal{X}_S \in \{0,1\}^t$ is the projection of $\mathcal{X}$ onto a random set $S \subseteq [n]$ of t positions, then, with high probability, $\mathcal{X}_S$ is statistically-close to a $\delta' t$-source, where $\delta' = \Omega(\delta)$. Whereas Nisan and Zuckerman [NZ96] and Vadhan [Vad04] were concerned with the amount of randomness that is required for sampling the t positions, in our case we can allow ourselves to sample the set S uniformly at random, and this leads to the following simplified form of the lemma:

Lemma 2.2 ([Vad04] – simplified)**.** *Let $\mathcal{X}$ be a δn-source over $\{0,1\}^n$, let $t \in [n]$, and let $\mathcal{S}$ denote the uniform distribution over sets $S \subseteq [n]$ of size t. Then, there exists a distribution $\mathcal{W}$ over $\{0,1\}^t$, jointly distributed with $\mathcal{S}$, such that the following hold:*

1. *$(\mathcal{S}, \mathcal{X}_{\mathcal{S}})$ is $2^{-\Omega(\delta t/\log(1/\delta))}$-close to $(\mathcal{S}, \mathcal{W})$.*
2. *For any set $S \subseteq [n]$ of size t it holds that $\mathcal{W}|_{\mathcal{S}=S}$ is a $\delta' t$-source for $\delta' = \delta/4$.*

2.2 Deterministic Public-Key Encryption

A deterministic public-key encryption scheme is almost identical to a (randomized) public-key encryption scheme, where the only difference is that the encryption algorithm is deterministic. More specifically, a deterministic public-key encryption scheme is a triple of polynomial-time algorithms $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$. The key-generation algorithm KG is a randomized algorithm which takes as input the security parameter 1^λ, where $\lambda \in \mathbb{N}$, and outputs a pair (pk, sk) of a public key pk and a secret key sk. The encryption algorithm Enc takes as input the security parameter 1^λ, a public key pk, and a plaintext $m \in \{0,1\}^{n(\lambda)}$, and outputs a ciphertext $c \in \{0,1\}^{t(\lambda)}$. The (possibly deterministic) decryption algorithm Dec takes as input the security parameter 1^λ, a secret key sk, and a ciphertext $c \in \{0,1\}^{t(\lambda)}$, and outputs either a plaintext $m \in \{0,1\}^{n(\lambda)}$ or the special symbol $\bot$. For succinctness, we will always assume 1^λ as an implicit input to all algorithms and refrain from explicitly specifying it.

In terms of security, in this paper we follow the standard approach for formalizing the security of deterministic public-key encryption schemes introduced by Bellare, Boldyreva and O'Neill [BBO07] and further studied by Bellare, Fischlin, O'Neill and Ristenpart [BFO+08a] and by Boldyreva, Fehr and O'Neill [BFO08b]. Specifically, we consider the PRIV-IND notion of security asking that any efficient algorithm has only a negligible advantage in distinguishing between encryptions of different sequences of plaintexts as long as each plaintext is sampled from high-entropy sources. We also consider the PRIV1-IND notion of security that focuses on a single plaintext, and asks that any efficient algorithm has only a negligible advantage in distinguishing between encryptions of different plaintexts that are sampled from high-entropy sources. This notion of security was shown by Boldyreva, Fehr and O'Neill [BFO08b] to guarantee security for block-sources of messages (that is, for sequences of messages where each message has high-entropy even when conditioned on the previous messages).

For defining these notions of security we rely on the following notation. We denote by $\mathbf{m} = (m_1, \ldots, m_\ell)$ a sequence of plaintexts, and by $\mathbf{c} = \mathsf{Enc}_{pk}(\mathbf{m})$ the sequence of their encryptions $(\mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell))$ under a public key pk.

Definition 2.3 (k-source ℓ-message adversary)**.** *Let $A = (A_1, A_2)$ be a probabilistic polynomial-time algorithm, and let $k = k(\lambda)$ and $\ell = \ell(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$. For any*

$\lambda \in \mathbb{N}$ *denote by* $(\mathcal{M}_\lambda^{(0)}, \mathcal{M}_\lambda^{(1)}, \mathcal{STATE}_\lambda)$ *the distribution corresponding to the output of* $A_1(1^\lambda)$. *Then,* A *is a* k-source ℓ-message adversary *if the following properties hold:*

1. $\mathcal{M}_\lambda^{(b)} = \left(\mathcal{M}_{1,\lambda}^{(b)}, \ldots, \mathcal{M}_{\ell,\lambda}^{(b)}\right)$ *is a distribution over sequences of* ℓ *plaintexts for each* $b \in \{0,1\}$.
2. *For any* $\lambda \in \mathbb{N}$, $i, j \in [\ell]$, *and for every triplet* $\left(\left(m_1^{(0)}, \ldots, m_\ell^{(0)}\right), \left(m_1^{(1)}, \ldots, m_\ell^{(1)}\right), \mathsf{state}\right)$ *that is produced by* $A_1(1^\lambda)$ *it holds that* $m_i^{(0)} = m_j^{(0)}$ *if and only if* $m_i^{(1)} = m_j^{(1)}$.
3. *For any* $\lambda \in \mathbb{N}$, $b \in \{0,1\}$, $i \in [\ell]$, *and* $\mathsf{state} \in \{0,1\}^*$ *it holds that* $\mathcal{M}_{i,\lambda}^{(b)}|_{\mathcal{STATE}_\lambda = \mathsf{state}}$ *is a* $k(\lambda)$*-source.*

Definition 2.4 (PRIV-IND)**.** *A deterministic public-key encryption scheme* $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is* PRIV-IND-secure for $k(\lambda)$-source $\ell(\lambda)$-message adversaries *if for any probabilistic polynomial-time* $k(\lambda)$*-source* $\ell(\lambda)$*-message adversary* $A = (A_1, A_2)$ *there exists a negligible function* $\nu(\lambda)$ *such that*

$$\mathsf{Adv}_{\Pi,A,\lambda}^{\mathsf{PRIV-IND}} \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV-IND}}(0) = 1\right] - \Pr\left[\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV-IND}}(1) = 1\right]\right| \leq \nu(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$, *where* $\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV-IND}}(b)$ *is defined as follows:*

1. $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$.
2. $(\mathbf{m}_0, \mathbf{m}_1, \mathsf{state}) \leftarrow A_1(1^\lambda)$.
3. $\mathbf{c} \leftarrow \mathsf{Enc}_{pk}(\mathbf{m}_b)$.
4. *Output* $A_2(1^\lambda, pk, \mathbf{c}, \mathsf{state})$.

Definition 2.5 (PRIV1-IND)**.** *A deterministic public-key encryption scheme* $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is* PRIV1-IND-secure for $k(\lambda)$-source adversaries *if for any probabilistic polynomial-time* $k(\lambda)$*-source* 1*-message adversary* $A = (A_1, A_2)$ *there exists a negligible function* $\nu(\lambda)$ *such that*

$$\mathsf{Adv}_{\Pi,A,\lambda}^{\mathsf{PRIV1-IND}} \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV1-IND}}(0) = 1\right] - \Pr\left[\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV1-IND}}(1) = 1\right]\right| \leq \nu(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$, *where* $\mathsf{Expt}_{\Pi,A,\lambda}^{\mathsf{PRIV1-IND}}(b)$ *is defined as follows:*

1. $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$.
2. $(m_0, m_1, \mathsf{state}) \leftarrow A_1(1^\lambda)$.
3. $c \leftarrow \mathsf{Enc}_{pk}(m_b)$.
4. *Output* $A_2(1^\lambda, pk, c, \mathsf{state})$.

2.3 The Decisional Diffie-Hellman Assumption

Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes as input a security parameter 1^λ, and outputs a triplet (G, p, g) where G is a group of prime order p that is generated by $g \in G$, and p is a λ-bit prime number. The Decisional Diffie-Hellman (DDH) assumption is that the ensembles $\{(G, g_1, g_2, g_1^r, g_2^r)\}_{\lambda \in \mathbb{N}}$ and $\{(G, g_1, g_2, g_1^{r_1}, g_2^{r_2})\}_{\lambda \in \mathbb{N}}$ are computationally indistinguishable, where $(G, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$, and the elements $g_1, g_2 \in G$ and $r, r_1, r_2 \in \mathbb{Z}_p$ are chosen independently and uniformly at random.

3 Modeling Incremental Deterministic Public-Key Encryption

In this section we present a framework for modeling the incrementality of deterministic public-key encryption schemes. Intuitively, a deterministic public-key encryption scheme is *incremental* if any small modification of a plaintext m resulting in a plaintext m' can be efficiently carried over for updating the encryption $c = \mathsf{Enc}_{pk}(m)$ of m to the encryption $c' = \mathsf{Enc}_{pk}(m')$ of m'. For capturing the efficiency of such an update operation we consider two natural complexity measures[4]:

- Input locality: The number of ciphertexts bits that are affected when flipping a single plaintext bit.
- Query complexity: The number of public-key, plaintext, and ciphertext bits that have to be read in order to update the ciphertext when flipping a single plaintext bit.

For capturing the above measures of efficiency we model the update operation as a probabilistic polynomial-time algorithm Update that receives as input the index i^* of a plaintext bit to be flipped, and has oracle access to the individual bits of the public key pk, the plaintext m to be modified, and to its encryption $c = \mathsf{Enc}_{pk}(m)$. That is, the algorithm Update can submit queries of the form (pk, i), (m, i) or (c, i), which are answered with the ith bit of pk, m, or c, respectively.

More formally, let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a deterministic public-key encryption scheme with message space $\{0,1\}^n$ and ciphertext space $\{0,1\}^t$ (where $n = n(\lambda)$ and $t = t(\lambda)$ are functions of the security parameter $\lambda \in \mathbb{N}$), and let Update be its corresponding update algorithm. We denote by $S \leftarrow \mathsf{Update}^{pk,m,c}(1^\lambda, i^*)$ the process in which the update algorithm with input $i^* \in [n]$ and oracle access to the individual bits of the public key pk, the plaintext m to be modified, and to its encryption $c = \mathsf{Enc}_{pk}(m)$, outputs a set $S \subseteq [t]$ of positions indicating which bits of the ciphertext c have to be flipped.

Definition 3.1 (Incremental deterministic PKE)**.** *Let* $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a deterministic public-key encryption scheme with message space* $\{0,1\}^n$ *and ciphertext space* $\{0,1\}^t$*, where* $n = n(\lambda)$ *and* $t = t(\lambda)$ *are functions of the security parameter* $\lambda \in \mathbb{N}$*. The scheme* Π *is* $\Delta(\lambda)$-incremental *is there exists a probabilistic polynomial-time algorithm* Update *satisfying the following requirements:*

1. *Correctness: There exists a negligible function* $\nu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, for any plaintext* $m \in \{0,1\}^n$ *and for any index* $i^* \in [n]$ *it holds that*

$$\Pr\left[c' = \mathsf{Enc}_{pk}(m') \;\middle|\; \begin{array}{c} c = \mathsf{Enc}_{pk}(m),\ S \leftarrow \mathsf{Update}^{pk,m,c}(1^\lambda, i^*) \\ m'[i^*] = \neg m[i^*] \text{ and } m'[i] = m[i] \text{ for all } i \in [n] \setminus \{i^*\} \\ c'[j] = \neg c[j] \text{ for all } j \in S \text{ and } c'[j] = c[j] \text{ for all } j \in [t] \setminus S \end{array} \right] \geq 1 - \nu(\lambda),$$

   where the probability is taken over the internal coin tosses of KG *and* Update*.*

2. *Efficiency: For all sufficiently large* $\lambda \in \mathbb{N}$ *the algorithm* $\mathsf{Update}^{(\cdot)}(1^\lambda, \cdot)$ *issues at most* $\Delta(\lambda)$ *oracle queries and outputs sets of size at most* $\Delta(\lambda)$*.*

Access to the plaintext. When providing the update algorithm with oracle access to the bits of the plaintext $m \in \{0,1\}^n$ we can assume without loss of generality that the only update operations are to flip the ith bit of m for $i \in [n]$. That is, one can also consider the operation of setting the ith bit of m to 0 or 1, but this can be handled by first querying the ith bit of m and then flipping it if it is different than the required value. We note, however, that for supporting only flipping operations it is not clear that access to the plaintext must be provided.

An important observation is that when access to the plaintext is not provided (i.e., when the update algorithm can query only the public key and the ciphertext), it is impossible to support the operation of setting a bit to 0 and 1 while providing PRIV1-IND security. That is, any such update

[4]For simplicity we focus on the case where both plaintexts and ciphertexts are represented as bit strings. We note, however, that our approach easily generalizes to arbitrary message and ciphertext spaces.

algorithm can be used to attack the PRIV1-IND security of the scheme by distinguishing between encryptions of high-entropy messages (and this holds for any level of incrementality)[5].

Privately-incremental schemes. In various scenarios it may be natural to provide the update algorithm with access not to the plaintext m but rather to the secret key sk (and thus indirect access to the plaintext which may be less efficient in terms of query complexity). Consider for example, a scenario in which a client stores an encrypted version $\bar{F}$ of a file F on a remote and untrusted server. In this the client does not have direct access to the file F, but only indirect access by using its secret key to recover parts of the file. In such a scenario it is required to capture the efficiency of the client by considering its query complexity to the secret key (and ciphertext) and not to the plaintext. This leads to a natural variant of Definition 3.1 in which the update algorithm is given oracle access to the public key pk, the secret key sk, and the ciphertext c (but no direct access to the plaintext).

Definition 3.2 (Privately-incremental deterministic PKE)**.** *Let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a deterministic public-key encryption scheme with message space $\{0,1\}^n$ and ciphertext space $\{0,1\}^t$, where $n = n(\lambda)$ and $t = t(\lambda)$ are functions of the security parameter $\lambda \in \mathbb{N}$. The scheme Π is $\Delta(\lambda)$-*privately-incremental *is there exists a probabilistic polynomial-time algorithm* Update *satisfying the following requirements:*

1. *Correctness: There exists a negligible function $\nu(\lambda)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, for any plaintext $m \in \{0,1\}^n$ and for any index $i^* \in [n]$ it holds that*

$$\Pr\left[c' = \mathsf{Enc}_{pk}(m') \;\middle|\; \begin{array}{c} c = \mathsf{Enc}_{pk}(m),\ S \leftarrow \mathsf{Update}^{pk,sk,c}(1^\lambda, i^*) \\ m'[i^*] = \neg m[i^*] \text{ and } m'[i] = m[i] \text{ for all } i \in [n] \setminus \{i^*\} \\ c'[j] = \neg c[j] \text{ for all } j \in S \text{ and } c'[j] = c[j] \text{ for all } j \in [t] \setminus S \end{array} \right] \geq 1 - \nu(\lambda),$$

   where the probability is taken over the internal coin tosses of KG *and* Update*.*

2. *Efficiency: For all sufficiently large $\lambda \in \mathbb{N}$ the algorithm $\mathsf{Update}^{(\cdot)}(1^\lambda, \cdot)$ issues at most $\Delta(\lambda)$ oracle queries and outputs sets of size at most $\Delta(\lambda)$.*

4 A Generic Construction via Random Partitioning

In this section we present a generic construction of an incremental PRIV1-IND-secure deterministic public-key encryption scheme from any PRIV-IND-secure deterministic public-key encryption scheme. As discussed in Section 1.3 our approach is a "randomized" alternative to the commonly-used approach of dividing the plaintext into small blocks and encrypting each block. Instead of dividing an n-bit plaintext m into fixed blocks, we project it onto a uniformly chosen partition $S_1, \ldots, S_{n/t}$ of the plaintext positions $\{1, \ldots, n\}$ to sets of size t each, and then separately encrypt each of the projections $m_{S_1}, \ldots, m_{S_{n/t}}$ using the underlying encryption scheme. Thus, when flipping a single bit of m we only need to update the encryption of the projection m_{S_i} for which the corresponding position belongs to the set S_i. Therefore, the resulting scheme enjoys the same incrementality that the underlying scheme has for small blocks. A more formal description follows.

[5] Consider the adversary $A = (A_1, A_2)$ that is defined as follows. The algorithm A_1 outputs $(m_0, m_1, \mathsf{state})$ where $m_0 \leftarrow \mathcal{U}_k || 0^{n-k}$ and $m_1 \leftarrow \mathcal{U}_n$ are sampled independently at random, and $\mathsf{state} = \bot$. That is, m_0 is a distributed uniformly conditioned on ending with 0^{n-k}, and m_1 is distributed uniformly. The algorithm A_2 on input $c = \mathsf{Enc}_{pk}(m_b)$ invokes the update algorithm to set the leftmost k bits of the plaintext corresponding to c to 0, and then compares the resulting ciphertext to $\mathsf{Enc}_{pk}(0^n)$. Note that if $b = 0$ then the two ciphertexts are always equal, and if $b = 1$ then they are equal only with probability $2^{-(n-k)}$.

The scheme. Let $\Pi' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ be a deterministic public-key encryption scheme for n'-bit plaintexts that is IND-PRIV-secure for k'-source ℓ'-message adversaries, where $n' = n'(\lambda)$, $k' = k'(\lambda)$ and $\ell' = \ell'(\lambda)$ are functions of the security parameter $\lambda \in \mathbb{N}$. We construct a deterministic public-key encryption scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ for n-bit plaintexts that is PRIV1-IND-secure for k-source adversaries, where $n = n(\lambda)$ and $k = k(\lambda)$ are functions of the security parameter $\lambda \in \mathbb{N}$ as follows:

- The algorithm KG on input the security parameter 1^λ samples $(pk', sk') \leftarrow \mathsf{KG}'(1^\lambda)$ together with a uniformly chosen partition $S_1, \ldots, S_{n/t}$ of $[n]$, where each set in the partition is of size $t = \Theta(\frac{n}{k} \cdot k')$. It then outputs $pk = (pk', S_1, \ldots, S_{n/t})$ and $sk = sk'$.[6]
- The algorithm $\mathsf{Enc}_{pk}(\cdot)$ on input a plaintext $m \in \{0,1\}^n$ outputs the ciphertext $(\mathsf{Enc}'_{pk'}(1||m_{S_1}), \ldots, \mathsf{Enc}'_{pk'}(n/t||m_{S_{n/t}}))$.
- The algorithm $\mathsf{Dec}_{sk}(\cdot)$ on input a ciphertext $(c_1, \ldots, c_{n/t})$ computes $m_{S_i} = \mathsf{Dec}'_{sk'}(c_i)$ for every $i \in [n/t]$, and outputs the plaintext m defined by the projections $m_{S_1}, \ldots, m_{S_{n/t}}$.

The incrementality of the scheme. When flipping a single bit of a plaintext m we only need to apply the update algorithm of the underlying scheme Π' to update a single output block $c_i = \mathsf{Enc}'_{pk'}(m_{S_i})$. The underlying scheme might have trivial incrementality and require to re-encrypt the whole block (which is significantly shorter than the length of the plaintext m), and in this case the update complexity is inherited from the efficiency of the encryption algorithm Enc' (below we discuss specific instantiations).

The security of the scheme. The main idea underlying the proof of security is that for a plaintext m that has min-entropy k, the "sample-and-extract" together with our choice of $t = \Theta(\frac{n}{k} \cdot k')$, and the fact that each set S_i is individually uniform imply that each of the encrypted strings $i||m_{S_i}$ is statistically-close to having min-entropy k'. The PRIV-IND security of Π' then immediately yields the PRIV1-IND security of Π. This enables us to prove the following theorem:

Theorem 4.1. *Assuming that Π' encrypts n'-bit plaintexts, for $n' = t + \log(n/t)$, and is IND-PRIV-secure for k'-source ℓ'-message adversaries, for some $k' = \omega(\log^2 n)$ and for $\ell' = n/t$, the scheme Π is PRIV1-IND-secure for k-sources.*

Proof. For any k-source adversary $A = (A_1, A_2)$ against PRIV1-IND security of the scheme Π we show that there exists an adversary $A' = (A'_1, A'_2)$ that is statistically close to a k'-source n/t-message adversary against the PRIV-IND security of the scheme Π' and has the same advantage.

The algorithm A'_1. On input 1^λ the algorithm A'_1 samples $\left(m^{(0)}, m^{(1)}, \mathsf{state}\right) \leftarrow A_1(1^\lambda)$ and a uniformly chosen partition $S_1, \ldots, S_{n/t}$ of $[n]$, where each set in the partition is of size $t = \Theta(\frac{n}{k} \cdot k')$. Then, it outputs $(\mathbf{m}_0, \mathbf{m}_1, \mathsf{state}')$, where $\mathbf{m}_b = \left(1||m^{(b)}_{S_1}, \ldots, n/t||m^{(b)}_{S_{n/t}}\right)$ for each $b \in \{0,1\}$, and $\mathsf{state}' = (\mathsf{state}, S_1, \ldots, S_{n/t})$.

The algorithm A'_2. On input $(1^\lambda, pk', \mathbf{c}, \mathsf{state}')$ the algorithm A'_2 first parses state' as $\mathsf{state}' = (\mathsf{state}, S_1, \ldots, S_{n/t})$ and defines $pk = (pk', S_1, \ldots, S_{n/t})$. Then, it outputs $A_2(1^\lambda, pk, \mathbf{c}, \mathsf{state})$.

Note that A' provides a perfect simulation of the $\mathsf{Expt}^{\mathsf{PRIV-IND}}_{\Pi,A,\lambda}(0)$ and $\mathsf{Expt}^{\mathsf{PRIV-IND}}_{\Pi,A,\lambda}(1)$ to A, and therefore we only need to prove that A' is statistically close to a k'-source n/t-message adversary. First, observe that in any vector of plaintexts $\mathbf{m}_b = \left(1||m^{(b)}_{S_1}, \ldots, n/t||m^{(b)}_{S_{n/t}}\right)$ that is produced by A_1 it always holds that all plaintexts are distinct (and this holds for both $b = 0$ and $b = 1$). Second,

[6] Without loss of generality we can assume that t divides n, as otherwise we can pad plaintexts with at most t zeros, and for our choice of parameters this would only have a minor effect on the min-entropy rate.

the fact that A is a k-source adversary means that for each $b = 0$ the plaintexts $m^{(b)}$ is sampled from a source with min-entropy at least k over $\{0,1\}^n$, even when conditioned on state. In turn, the "sample-then-extract" lemma (see Lemma 2.2) that for each $b \in \{0,1\}$ and $i \in [n/t]$ the projection $m_{S_i}^{(b)}$ is $2^{-\Omega(\delta t/\log(1/\delta))}$-close to a source with min-entropy $\delta' t$ over $\{0,1\}^t$, where $\delta' = k/(4n)$. Our choice of $t = \Theta(\frac{n}{k} \cdot k')$ and the requirement $k' = \omega(\log^2 n)$ in the statement of the theorem imply that $m_{S_i}^{(b)}$ is $2^{-\omega(\log n)}$-close to a source with min-entropy k'. ■

Specific instantiations. By instantiating our generic construction with the two PRIV-IND-secure schemes of Bellare at el. [BBO07] in the random oracle model, we obtain schemes with essentially optimal incrementality $\Delta = O(t) = O(\frac{n}{k} \cdot \log^3 n)$. Their first scheme is based on any semantically-secure (randomized) public-key encryption scheme, and their second scheme is length-preserving based on RSA-OAEP. We note that when instantiating our generic construction with their length-preserving scheme there is in fact no need to concatenate the block number i to each projection m_{S_i} (for ensuring that we always encrypt distinct values), but only to use the block number as a prefix for the random oracle when encrypting m_{S_i}. Therefore in this case the resulting scheme is still a length-preserving one.

5 A Construction Based on the Decisional Diffie-Hellman Assumption

In this section we construct a deterministic public-key encryption scheme that enjoys essentially optimal incrementality, and guarantees PRIV1-IND security based on the Decisional Diffie-Hellman (DDH) assumption. We begin by introducing rather standard notation and then describe the scheme.

Notation. Let G be a group of prime order p that is generated by $g \in G$. For any matrix $A = \{a_{ij}\}_{i\in[n],j\in[n]} \in \mathbb{Z}_p^{n\times n}$ we denote by $g^A \in G^{n\times n}$ the matrix $\{g^{a_{ij}}\}_{i\in[n],j\in[n]}$. In addition, for a column vector $m = (m_1, \ldots, m_n)^\mathsf{T} \in \mathbb{Z}_p^n$ and a matrix $A = \{a_{ij}\}_{i\in[n],j\in[n]} \in \mathbb{Z}_p^{n\times n}$ we define

$$A \odot g^m \stackrel{\text{def}}{=} g^A \odot m \stackrel{\text{def}}{=} g^{Am} = (g^{\sum_i a_{1,i} m_i}, \ldots, g^{\sum_i a_{n,i} m_i})^\mathsf{T} \in G^n \quad .$$

The scheme. Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes as input the security parameter 1^λ, and outputs a triplet (G, p, g) where G is a group of prime order p that is generated by $g \in G$, and p is a λ-bit prime number. The scheme is parameterized by the security parameter λ, the message length $n = n(\lambda)$, and the min-entropy $k = k(\lambda)$ for which the scheme is secure. Both n and k are polynomials in the security parameter. The scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is defined as follows:

- **Key generation.** The algorithm KG on input the security parameter 1^λ samples $(G, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$, and a matrix $A \leftarrow \mathcal{A}_{n,k,p}$, where $\mathcal{A}_{n,k,p}$ is a distribution over $\mathbb{Z}_p^{n\times n}$ which is defined below. It then outputs $pk = (G, p, g, g^A)$ and $sk = A^{-1}$.
- **Encryption.** The algorithm $\mathsf{Enc}_{pk}(\cdot)$ on input a plaintext $m \in \{0,1\}^n$ outputs the ciphertext $g^A \odot m = g^{Am} \in G^n$.
- **Decryption.** The algorithm $\mathsf{Dec}_{sk}(\cdot)$ on input a ciphertext $g^c = (g^{c_1}, \ldots, g^{c_n}) \in G^n$ first computes $w = A^{-1} \odot g^c = g^{A^{-1}c} \in G^n$, and lets $w = (g^{m_1}, \ldots, g^{m_n})$. If $m = (m_1, \ldots, m_n) \in \{0,1\}^n$ (note that this test can be computed efficiently) then it outputs m, and otherwise it outputs $\perp$.

The distribution $\boldsymbol{\mathcal{A}_{n,k,p}}$. For completing the description of our scheme it remains to specify the distribution $\mathcal{A}_{n,k,p}$ that is defined over $\mathbb{Z}_p^{n\times n}$. Looking ahead this distribution will be used to define the distribution of injective keys in our collection of smooth trapdoor functions. In fact, we find it convenient to first specify the distribution $\widetilde{\mathcal{A}}_{n,k,p}$ that will be used to define the distribution of smooth keys. These two distributions rely on the following distributions as building block:

- $\boldsymbol{\mathcal{R}_{n,k,p}}$**: sparse random $\boldsymbol{\ell \times n}$ matrices.** The distribution $\mathcal{R}_{n,k,p}$ is defined as a random sample from $\mathbb{Z}_p^{\ell \times n}$ matrices that have exactly $t = \Theta(\frac{n}{k} \cdot \log^3 n)$ non-zero entries in each row, where $\ell = \Theta(k/\log p)$.
- $\boldsymbol{\mathcal{D}_{n,k,p}}$**: diagonally-striped $\boldsymbol{\ell \times n}$ matrices.** The distribution $\mathcal{D}_{n,k,p}$ is defined as a random sample from $\mathbb{Z}_p^{\ell \times n}$ matrices whose elements d_{ij} are non-zero if and only if $i \equiv j \pmod{\ell}$ (for simplicity we assume that n is divisible by ℓ).

The distribution $\widetilde{\mathcal{A}}_{n,k,p}$ over $\mathbb{Z}_p^{n \times n}$ is defined as matrices $\widetilde{A}$ obtained by independently sampling $R \leftarrow \mathcal{R}_{n,k,p}$, $D_1 \leftarrow \mathcal{D}_{n,k,p}$, and $D_2 \leftarrow \mathcal{D}_{n,k,p}$, and letting $\widetilde{A} \stackrel{\text{def}}{=} D_2^{\mathsf{T}} \times (R + D_1)$. Then, the distribution $\mathcal{A}_{n,k,p}$ is defined as matrices A obtained by sampling a matrix $\widetilde{A} \leftarrow \widetilde{\mathcal{A}}_{n,k,p}$ and then *re-sampling* all its non-zero entries from $\mathbb{Z}_p$ independently and uniformly at random. In other words, the resulting matrix A preserves zeroes of the matrix $\widetilde{A}$, while randomizing all other elements (and thus linear dependencies between rows) of the original matrix. See Figure 1 for an illustration of the distributions $\mathcal{R}_{n,k,p}$, $\mathcal{D}_{n,k,p}$ and $\widetilde{\mathcal{A}}_{n,k,p}$.

Intuitively, the matrix D_1 is only meant to ensure that such the resulting matrix A is invertible. Indeed, the matrix D_1 guarantees that with an overwhelming probability all the elements on the main diagonal of A are non-zeros. Now, ignoring the matrix D_1, the matrix $\widetilde{A}$ is generated to satisfy two properties. The first is that each of its first ℓ rows has t randomly chosen entries with values that are chosen uniformly from $\mathbb{Z}_p$, and all other $n - t$ entries are zeros. Looking ahead, when computing the inner product of such a row with a source of min-entropy larger than $\log p$, the "sample-then-extract" lemma (see Lemma 2.2) guarantees that the output is statistically close to uniform. The second property, is that each of its last $n - \ell$ rows are linear combinations of the first ℓ rows, and therefore the image of its corresponding linear map is determined by the first ℓ rows.

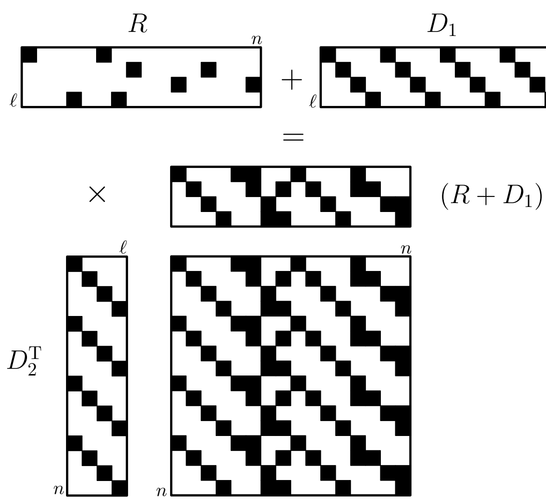

Figure 1: The distributions $\mathcal{R}_{n,k,p}$, $\mathcal{D}_{n,k,p}$ and $\widetilde{\mathcal{A}}_{n,k,p}$.

The incrementality of the scheme. When naturally storing the public-key element g^A as a sparse matrix, listing only the entries corresponding to the non-zero entries of A, the incrementality of the scheme corresponds to the maximal number of non-zero entries in the columns of A (up to a multiplicative $\log p$ factor that needs to be taken into account since we measure incrementality by *bit* operations). It is easy to see that for our choice of $t = \Theta(\frac{n}{k} \cdot \log^3 n)$ and $\ell = \Theta(k/\log p)$, each column of the matrix A has at most $O(t + n/\ell) = O(\frac{n}{k} \cdot \log^3 n)$ non-zero entries with all but a negligible probability.

The security of the scheme. As discussed in Section 1.3, the security of our scheme is based on the notion of *smooth trapdoor functions* which we formalize in Section 5.1, and then in Section 5.2 we show that our scheme is in fact a collection of smooth trapdoor functions. This enables us to prove the following theorem:

Theorem 5.1. *Under the Decisional Diffie-Hellman assumption the scheme* Π *is PRIV1-IND-secure for* k*-sources.*

5.1 Smooth Trapdoor Functions

A *collection of smooth trapdoor functions* consists of two families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor. Functions in the other family are "smooth" in the sense that their output distribution on any source of input with high min-entropy is statistically close to their output distribution on a uniformly sampled input. The only security requirement is that a description of a randomly chosen function from the family of injective functions is computationally indistinguishable from a description of a randomly chosen function from the family of smooth functions.

Definition 5.2 (Smooth trapdoor functions)**.** *Let* $n = n(\lambda)$ *and* $k = k(\lambda)$ *be functions of the security parameter* $\lambda \in \mathbb{N}$*. A* collection of (n, k)-smooth trapdoor functions *is a* 4*-tuple of probabilistic polynomial-time algorithms* $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$ *such that:*

1. *Injectivity: With overwhelming probability over the choice of* $(pk, sk) \leftarrow \mathsf{KG}_{\mathsf{Inj}}(1^\lambda)$*, for every* $x \in \{0,1\}^n$ *it holds that* $\mathsf{F}^{-1}_{sk}(\mathsf{F}_{pk}(x)) = x$*.*
2. *Smoothness: For every* k*-source* $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ *over* $\{0,1\}^n$ *the statistical distance between the distributions* $\{\mathsf{F}_{pk}(x) \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda), x \leftarrow \mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathsf{F}_{pk}(x) \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda), x \leftarrow \mathcal{U}_n\}_{\lambda\in\mathbb{N}}$ *is negligible in* λ*.*
3. *Indistinguishability: The two distributions* $\{pk \colon (pk, sk) \leftarrow \mathsf{KG}_{\mathsf{Inj}}(1^\lambda)\}_{\lambda\in\mathbb{N}}$ *and* $\{pk \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda)\}_{\lambda\in\mathbb{N}}$ *are computationally indistinguishable.*

We note that the definition of a *hidden universal-mode encryption* of Boldyreva, Fehr and O'Neill [BFO08b] is stronger than our definition of smooth trapdoor functions, as evident from our construction in this section which is not universal in its smooth mode (that would interfere with the incrementality requirement). In addition, Boldyreva et al. showed that the composition of any lossy trapdoor function with a pairwise independent permutation is a hidden universal-mode encryption, and thus a collection of smooth trapdoor functions. The pairwise independent permutation, however, again contradicts the incrementality property that we require.

The following theorem states that any collection of smooth trapdoor functions is also a PRIV1-IND-secure deterministic public-key encryption scheme. The theorem was implicitly proved by Boldyreva et al. [BFO08b, Theorem 5.1], and here we provide its proof for completeness in light of our new notion of smooth trapdoor functions.

Theorem 5.3. *Let* $n = n(\lambda)$ *and* $k = k(\lambda)$ *be functions of the security parameter* $\lambda \in \mathbb{N}$*, and let* $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$ *be a collection of* (n, k)*-smooth trapdoor functions. Then* $\Pi = (\mathsf{KG}_{\mathsf{Inj}}, \mathsf{F}, \mathsf{F}^{-1})$ *is a deterministic public-key encryption scheme that is PRIV1-IND-secure for* k*-sources.*

Proof. Let $A = (A_1, A_2)$ be a k-source adversary. For any $\lambda \in \mathbb{N}$ and $b \in \{0,1\}$ we denote by $\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(b)$ the experiment that is obtained from the experiment $\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(b)$ by sampling the public key pk using $\mathsf{KG}_{\mathsf{Smooth}}(1^\lambda)$ instead of $\mathsf{KG}_{\mathsf{Inj}}(1^\lambda)$. Then,

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda} &= \left|\Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right]\right| \\
&\leq \left|\Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right] - \Pr\left[\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right]\right| \qquad (5.1) \\
&\quad + \left|\Pr\left[\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right] - \Pr\left[\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right]\right| \qquad (5.2) \\
&\quad + \left|\Pr\left[\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right]\right| . \qquad (5.3)
\end{aligned}$$

By definition for any $b \in \{0,1\}$ the experiments $\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(b)$ and $\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(b)$ differ only on the distribution of the public key pk. Therefore, the indistinguishability property of the collection $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$ between public keys that are "injective" and "smooth" directly guarantees that the terms in Equations (5.1) and (5.3) are negligible.

In addition, the smoothness property of the collection $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$ and the fact that A is a k-source adversary guarantee that in the experiments $\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0)$ and $\widetilde{\mathsf{Expt}}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1)$ the ciphertext $c = \mathsf{Enc}_{pk}(m_b)$ is statistically-close to the encryption of a uniformly distributed message from A's point of view. This implies that also the term in Equation (5.2) is negligible, and concludes the proof of the theorem. ■

5.2 Proof of Security

The description of our encryption scheme naturally defines a 4-tuple $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$, which we show here to be a collection of smooth trapdoor functions. Specifically, the algorithms $\mathsf{KG}_{\mathsf{Inj}}$, F, and F^{-1} are the key-generation, encryption and decryption algorithms of our encryption scheme, respectively, and the algorithm $\mathsf{KG}_{\mathsf{Smooth}}$ is the key-generation algorithm that uses the distribution $\widetilde{\mathcal{A}}_{n,k,p}$ (instead of $\mathcal{A}_{n,k,p}$). The security of our encryption scheme (i.e., Theorem 5.1) then follows as a corollary by putting together Theorem 5.3 and the following theorem:

Theorem 5.4. *Under the Decisional Diffie-Hellman assumption,* $(\mathsf{KG}_{\mathsf{Inj}}, \mathsf{KG}_{\mathsf{Smooth}}, \mathsf{F}, \mathsf{F}^{-1})$ *is a collection of* (n,k)*-smooth trapdoor functions.*

Proof. We prove the theorem using the following three lemmas, establishing the required properties of injectivity, smoothness, and indistinguishability.

Lemma 5.5 (Injectivity)**.** *With overwhelming probability over the choice of* $(pk, sk) \leftarrow \mathsf{KG}_{\mathsf{Inj}}(1^\lambda)$*, for every* $x \in \{0,1\}^n$ *it holds that* $\mathsf{F}^{-1}_{sk}(\mathsf{F}_{pk}(x)) = x$*.*

Proof of Lemma 5.5. We prove the lemma by showing that a matrix drawn from $\mathcal{A}_{n,k,p}$ is invertible except with probability $O(n/p)$. Consider the intermediate steps of drawing a matrix A from $\mathcal{A}_{n,k,p}$. First R, D_1 and D_2 are sampled from $\mathcal{R}_{n,k,p}$, $\mathcal{D}_{n,k,p}$, and $\mathcal{D}_{n,k,p}$ respectively. Then the matrix $\widetilde{A}$ is computed as $A = D_2^{\mathsf{T}} \times (R + D_1)$, and the matrix A is produced by re-sampling all its non-zero entries. We show that A is invertible by arguing that all elements on its main diagonal are non-zero, except with probability $O(n/p)$.[7]

[7] This is well-known to imply invertibility: Express $\det A$ as a function of formal variables corresponding to the non-zero elements of A. The total degree of this polynomial is n, and since the main diagonal of A is non-zero, the polynomial is not identically zero. By the Schwartz-Zippel lemma, the probability that this polynomial evaluates to zero (and thus the matrix is rank-deficient) is $O(n/p)$.

Call the elements of an $\ell \times n$ matrix with coordinates (u, v), where $u \equiv v \pmod{\ell}$, *pseudodiagonal.* The pseudodiagonal elements of D_2 are non-zero by construction. There are exactly n pseudodiagonal elements chosen at random in D_1, and with probability $O(n/p)$ one of them is zero. The probability that any of the pseudodiagonal elements of D_1 is canceled after summing it with R is $O(n/p)$. Conditioning on these events *not* happening, i.e., all pseudodiagonal elements of $D_1 + R$ are not zero, all elements on the main diagonal of M that are products of two pseudodiagonal elements from $D_1 + R$ and D_2, are thus also non-zero.

It means that when the non-zero entries of the matrix $\widetilde{A}$ are re-sampled to produce A, all elements on the main diagonal of A will be assigned fresh random values from $\mathbb{Z}_p$, and thus they will all be non-zero except with probability $O(n/p)$. ∎

Lemma 5.6 (Smoothness)**.** *For every k-source $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ over $\{0,1\}^n$ the statistical distance between the distributions $\{\mathsf{F}_{pk}(x) \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda), x \leftarrow \mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{F}_{pk}(x) \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda), x \leftarrow \mathcal{U}_n\}_{\lambda\in\mathbb{N}}$ is negligible in λ.*

Proof of Lemma 5.6. Fix $\lambda \in \mathbb{N}$, and let $\widetilde{A} \leftarrow \widetilde{\mathcal{A}}_{n,k,p}$, $x \leftarrow \mathcal{X}_\lambda$, and $y = Ax$. The first observation is that any matrix $\widetilde{A}$ sampled from $\widetilde{\mathcal{A}}_{n,k,p}$ has rank ℓ as a product of two matrices, each having rank at most ℓ. Further, for a given $\widetilde{A}$, the first ℓ entries of y determine the rest of y. Indeed, if i and j are two coordinates and $i \equiv j \pmod{\ell}$, then y_i/y_j is exactly the ratio of the corresponding (non-zero) elements of D_2 with coordinates $(i \bmod \ell, i)$ and $(j \bmod \ell, j)$ (identifying rows 0 and ℓ of D_2).

Therefore, it is sufficient to consider the distribution of $y = Ax$ over the first ℓ coordinates of the result, denoted as $\mathcal{Y}_1, \dots, \mathcal{Y}_\ell$. We shall prove that $\mathcal{Y}_1, \dots, \mathcal{Y}_\ell$ is statistically close to the uniform distribution over $\mathbb{Z}_p^\ell$ (i.e., independent of $\mathcal{X}$, and thus the lemma easily follows). Specifically, for every $i \in [\ell]$ we prove that with an overwhelming probability over the choice of $(y_1, \dots, y_{i-1}) \leftarrow (\mathcal{Y}_1, \dots, \mathcal{Y}_{i-1})$, it holds that the distribution of $\mathcal{Y}_i$ when conditioned on $\mathcal{Y}_1 = y_1, \dots, \mathcal{Y}_{i-1} = y_{i-1}$ is statistically close to the uniform distribution over $\mathbb{Z}_p$. A standard hybrid argument implies the claim about the joint distribution of $\mathcal{Y}_1, \dots, \mathcal{Y}_\ell$.

Recall that $\mathcal{X}$ is a source of min-entropy k. Lemma 2.1 guarantees that with probability $1 - 2^{-\log^2\lambda}$ over the choice of $(y_1, \dots, y_{i-1}) \leftarrow (\mathcal{Y}_1, \dots, \mathcal{Y}_{i-1})$, the min-entropy of $\mathcal{X}$ is at least $k - (i-1)\log p - \log^2\lambda$. Consider the evaluation of $y_i = \langle \widetilde{A}^{(i)}, x\rangle$ for $i \le \ell$, where $\widetilde{A}^{(i)}$ is the ith row of $\widetilde{A}$. By construction, $\widetilde{A}^{(i)} = d_i(R^{(i)} + D_1^{(i)})^\mathsf{T}$, where d_i is a pseudodiagonal element of D_2. Let S be the set of t non-zero entries of $R^{(i)}$ (the ith row of R).

By Lemma 2.2, the projection of $\mathcal{X}$ onto S, denoted as $\mathcal{X}_S$, is $2^{-\Omega(\delta t/\log(1/\delta))}$-close to a $k'/4$-source on t bits W, where $\delta = k'/n$. The action of $\widetilde{A}_S^{(i)}$ on W is the scalar product of a $k'/4$-source with a random vector (we account for the entries of $\widetilde{A}_S^{(i)}$ being non-zero by adding another t/p term to the statistical distance), which is a universal hash function from $\{0,1\}^t$ to $\mathbb{Z}_p$. By the leftover hash lemma, noting that $k' > k - \ell\log p - \log^2\lambda > k/2 > 2\log p$ the statistical distance between W and the uniform distribution on $\mathbb{Z}_p$ given $S, y_1, \dots, y_{i-1}$ is negligible. ∎

Lemma 5.7 (Indistinguishability)**.** *Under the Decisional Diffie-Hellman assumption, the distributions $\{pk \colon (pk, sk) \leftarrow \mathsf{KG}_{\mathsf{Inj}}(1^\lambda)\}_{\lambda\in\mathbb{N}}$ and $\{pk \colon pk \leftarrow \mathsf{KG}_{\mathsf{Smooth}}(1^\lambda)\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable.*

Proof of Lemma 5.7. Consider a sample from the distribution $\widetilde{\mathcal{A}}_{n,k,p}$. It is obtained by sampling a sparse $\ell \times n$ matrix (from the distribution $\mathcal{R}_{n,k,p} + \mathcal{D}_{n,k,p}$), and then replicating every row of this matrix n/ℓ times multiplying it with a random non-zero field element each time. The distribution

$\mathcal{A}_{n,k,p}$ is sampled by drawing a matrix from $\widetilde{\mathcal{A}}_{n,k,p}$ and re-sampling all its non-zero elements. Therefore, matrix minors defined as all non-zero elements of rows congruent modulo ℓ have rank 1 if the matrix is drawn from $\widetilde{\mathcal{A}}_{n,k,p}$ and rank n/ℓ if it comes from $\mathcal{A}_{n,k,p}$.

We use this observation together with the matrix-DDH assumption to prove the lemma. The matrix-DDH assumption, due to Boneh, Halevi, Hamburg, and Ostrovsky [BHH+08], asserts that $g^{\mathcal{A}} \stackrel{c}{\approx} g^B$, where distributions $\mathcal{A}$ and $\mathcal{B}$ are random matrices from $\mathbb{Z}_p^{a\times b}$ of ranks r_1 and r_2 respectively and $1 < r_1 < r_2 \leq \min(a,b)$. Boneh et al. prove that the DDH assumption implies matrix-DDH [BHH+08, Lemma 3.3].

Consider hybrid distributions $\mathcal{H}_0,\ldots,\mathcal{H}_\ell$, where $\mathcal{H}_0 = g^{\widetilde{\mathcal{A}}_{n,k,p}}$ and $\mathcal{H}_\ell = g^{\mathcal{A}_{n,k,p}}$. Each intermediate distribution $\mathcal{H}_i$ is obtained by drawing a matrix from $\widetilde{\mathcal{A}}_{n,k,p}$ and re-sampling all rows congruent to an element of the set $\{0,\ldots,i-1\}$ modulo ℓ (if $i = 0$, no rows are re-sampled).

The difference between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ is in the distribution of rows congruent to i modulo ℓ. We now change the procedure for sampling from $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ by embedding an instance of the matrix-DDH problem. Draw a matrix from $\mathcal{H}_i$. Let the number of entries not equal to 1 in the ith row be r. Sample a random rank-1 matrix A from $\mathbb{Z}_p^{n/\ell\times r}$. Replace the minor of $\mathcal{H}_i$ corresponding to the entries not equal to 1 in the rows congruent to i modulo ℓ with g^A. Analogously, change the distribution $\mathcal{H}_{i+1}$ by replacing the similarly defined minor with the matrix g^B, where B is a random n/ℓ-rank matrix of size $n/\ell \times r$. It is easy to check that except with probability $O(n^2/p)$ (to account for a possibility of zero elements in A or B) the new sampling procedures do not change the distributions $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$.

By the matrix-DDH assumption the resulting distributions are computationally indistinguishable. Applying the hybrid argument to the sequence $\mathcal{H}_0,\ldots,\mathcal{H}_\ell$, we complete the proof. ■

This settles the proof of Theorem 5.4. ■

6 The Lower Bound

In this section we prove a lower bound on the incrementality of deterministic public-key encryption schemes. More specifically, we prove a lower bound on the *input locality* of the encryption algorithm (recall that our notion of incrementality in Definition 3.1 considers in particular input locality). Recall that the input locality of a function $f : \{0,1\}^n \to \{0,1\}^t$ is the maximal number of output bits on which an input bit of f has influence[8]. We prove the following theorem:

Theorem 6.1. *Let $n = n(\lambda)$, $t = t(\lambda)$, and $k = k(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$, and let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a deterministic public-key encryption scheme with plaintext space $\{0,1\}^n$ and ciphertext space $\{0,1\}^t$. If Π is PRIV1-IND-secure for k-sources then for all sufficiently large $\lambda \in \mathbb{N}$ the function $\mathsf{Enc}_{pk} : \{0,1\}^n \to \{0,1\}^t$ has input locality at least $\frac{n-3}{k\log t}$ with probability at least $1/2$ over the choice of the public key pk.*

Proof. Assume towards a contradiction that for infinitely many $\lambda \in \mathbb{N}$ there exists a set $\mathcal{P}_\lambda$ of public keys such that: (1) $\Pr_{(pk,sk)\leftarrow\mathsf{KG}(1^\lambda)}[pk \in \mathcal{P}_\lambda] > 1/2$, and (2) Enc_{pk} has input locality $\Delta(\lambda) < \frac{n(\lambda)-3}{k(\lambda)\log t(\lambda)}$ for any $pk \in \mathcal{P}_\lambda$.

Consider the adversary $A = (A_1, A_2)$ that is defined as follows. The algorithm A_1 on input 1^λ outputs $(m_0, m_1, \mathsf{state})$ where $m_0 \leftarrow \mathcal{U}_k||0^{n-k}$ and $m_1 \leftarrow \mathcal{U}_n$ are sampled independently at random, and $\mathsf{state} = \perp$. That is, m_0 is a distributed uniformly conditioned on ending with 0^{n-k}, and m_1 is

[8] We say that the ith input bit of f *influences* the jth output bit if there exists an assignment to the input bits such that flipping the ith input bit changes the value of the jth output bit.

distributed uniformly. The algorithm A_2 on input a public key pk and a ciphertext $c = \mathsf{Enc}_{pk}(m_b)$ first computes $c^* = \mathsf{Enc}_{pk}(0^n)$. Then, if the Hamming distance between c and c^* is at most $k\Delta$ then it outputs 0, and otherwise it outputs 1. We now analyze the advantage of A by considering the cases $b = 0$ and $b = 1$.

The case $b = 0$. In this case the Hamming distance between m_0 and 0^n is at most k. For any $pk \in \mathcal{P}_\lambda$ each plaintext bit may affect at most Δ ciphertext bits, and therefore the Hamming distance between $c = \mathsf{Enc}_{pk}(m_0)$ and $c^* = \mathsf{Enc}_{pk}(0^n)$ is at most $k\Delta$. Thus, for any $pk \in \mathcal{P}_\lambda$ the adversary A will always output 0. This implies that for infinitely many $\lambda \in \mathbb{N}$ it holds that

$$\Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right] \leq \Pr_{(pk,sk)\leftarrow\mathsf{KG}(1^\lambda)}[pk \notin \mathcal{P}_\lambda] < 1/2 \ . \tag{6.1}$$

The case $b = 1$. In this case we prove an upper bound on the probability that the Hamming distance between $c = \mathsf{Enc}_{pk}(m_1)$ and $c^* = \mathsf{Enc}_{pk}(0^n)$ is at most $k\Delta$. The encryption algorithm outputs t-bit ciphertexts, and note that the number of t-bit strings that are within Hamming distance $k\Delta$ to c^* is at most

$$2^{k\Delta}\binom{t}{k\Delta} \leq (2t)^{k\Delta} \ .$$

As a result, the number of n-bit plaintexts whose ciphertext under pk is within Hamming distance $k\Delta$ to c^* is also at most $(2t)^{k\Delta}$. The plaintext m_1 is sampled uniformly at random from $\{0,1\}^n$, and therefore

$$\Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right] \geq 1 - \frac{(2t)^{k\Delta}}{2^n} \ . \tag{6.2}$$

This implies that on one hand, by combining Equations (6.1) and (6.2), for infinitely many $\lambda \in \mathbb{N}$ it holds that

$$\mathsf{Adv}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda} = \left|\Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda}(1) = 1\right]\right| > \frac{1}{2} - \frac{(2t)^{k\Delta}}{2^n} \ .$$

On the other hand, however, the PRIV1-IND security of the scheme guarantees that there is a negligible function $\nu = \nu(\lambda)$ (corresponding to the adversary A) such that $\mathsf{Adv}^{\mathsf{PRIV1-IND}}_{\Pi,A,\lambda} \leq \nu$ for all sufficiently large $\lambda \in \mathbb{N}$. Therefore,

$$\frac{1}{2} - \frac{(2t)^{k\Delta}}{2^n} \leq \nu \ ,$$

which implies that

$$\Delta \geq \frac{n-3}{k \log t} \ .$$

and yields a contradiction. ∎

Acknowledgements. We thank Salil Vadhan for useful discussions regarding Lemma 2.2.

References

[AIK06] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. *SIAM Journal on Computing*, 36(4):845–888, 2006.

[BBN$^+$09] M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In *Advances in Cryptology – ASIACRYPT '09*, pages 232–249, 2009.

[BBO07] M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In *Advances in Cryptology – CRYPTO '07*, pages 535–552, 2007.

[BCG$^+$00] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur. Single instance storage in Windows 2000. In *Proceedings of the 4th USENIX Windows Systems Symposium*, pages 13–24, 2000.

[BFO$^+$08a] M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *Advances in Cryptology – CRYPTO '08*, pages 360–378, 2008.

[BFO08b] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *Advances in Cryptology – CRYPTO '08*, pages 335–359, 2008.

[BGG94] M. Bellare, O. Goldreich, and S. Goldwasser. Incremental cryptography: The case of hashing and signing. In *Advances in Cryptology – CRYPTO '94*, pages 216–233, 1994.

[BGG95] M. Bellare, O. Goldreich, and S. Goldwasser. Incremental cryptography and application to virus protection. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, pages 45–56, 1995.

[BHH$^+$08] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from Decision Diffie-Hellman. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.

[BKY01] E. Buonanno, J. Katz, and M. Yung. Incremental unforgeable encryption. In *Proceedings of the 8th International Workshop on Fast Software Encryption*, pages 109–124, 2001.

[BM97] M. Bellare and D. Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In *Advances in Cryptology – EUROCRYPT '97*, pages 163–192, 1997.

[BS11] Z. Brakerski and G. Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In *Advances in Cryptology – CRYPTO '11*, pages 543–560, 2011.

[DAB$^+$02] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 617–624, 2002.

[DOR$^+$08] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[DS05] Y. Dodis and A. Smith. Entropic security and the encryption of high entropy messages. In *Proceedings of the 2nd Theory of Cryptography Conference*, pages 556–577, 2005.

[FGK$^+$10] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. In *Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*, pages 279–295, 2010.

[Fis97a] M. Fischlin. Incremental cryptography and memory checkers. In *Advances in Cryptology – EUROCRYPT '97*, pages 293–408, 1997.

[Fis97b] M. Fischlin. Lower bounds for the signature size of incremental schemes. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 438–447, 1997.

[FOR12] B. Fuller, A. O'Neill, and L. Reyzin. A unified approach to deterministic encryption: New constructions and a connection to computational entropy. Cryptology ePrint Archive, Report 2012/005, 2012.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[Mic97] D. Micciancio. Oblivious data structures: Applications to cryptography. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 456–464, 1997.

[NZ96] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 187–196, 2008.

[QD02] S. Quinlan and S. Dorward. Venti: A new approach to archival storage. In D. D. E. Long, editor, *FAST*, pages 89–101. USENIX, 2002.

[RW02] A. Russell and H. Wang. How to fool an unbounded adversary with a short key. In *Advances in Cryptology – EUROCRYPT '02*, pages 133–148, 2002. Full version is available as [RW06].

[RW06] A. Russell and H. Wang. How to fool an unbounded adversary with a short key. *IEEE Transactions on Information Theory*, 52(3):1130–1140, 2006.

[Vad04] S. P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Jounal of Cryptology*, 17(1):43–77, 2004.

[Wee12] H. Wee. Dual projective hashing and its applications - lossy trapdoor functions and more. To appear in *Advances in Cryptology – EUROCRYPT '12*, 2012.

[ZLP08] B. Zhu, K. Li, and R. H. Patterson. Avoiding the disk bottleneck in the data domain deduplication file system. In M. Baker and E. Riedel, editors, *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, pages 269–282. USENIX, 2008.